DESIGN AND IMPLEMENTATION OF A PERSONNEL MANAGEMENT INFORMATION SYSTEM

BY

AKINRIMISI OLADAYO EFEMENA
PSC1814735

DEPARTMENT OF COMPUTER SCIENCE
FACULTY OF COMPUTING
UNIVERSITY OF BENIN, BENIN CITY.
EDO STATE.

JANUARY, 2026

DESIGN AND IMPLEMENTATION OF A PERSONNEL MANAGEMENT INFORMATION SYSTEM

BY

AKINRIMISI OLADAYO EFEMENA
PSC1814735

A PROJECT WORK SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE, FACULTY OF COMPUTING UNIVERSITY OF BENIN, BENIN CITY, IN PARTIAL FULFILMENT OF THE AWARD OF BACHELOR OF SCIENCE (B.S.C HONOURS) DEGREE IN COMPUTER SCIENCE

JANUARY, 2026

CERTIFICATION

This is to certify that the project work was carried out by **AKINRIMISI OADAYO EFEMENA** with matriculation number **PSC1814735** to the Department of Computer Science, Faculty of Computing in partial fulfillment for the Requirement for the award of the Bachelor of Sciences(B.S c.) Degree in the Department of Computer Science, University of Benin, Benin City, Edo State.

_____ _____

DR. (MRS) R.O. OSASERI **Signature/Date**
Project Supervisor

APPROVAL

This research project is hereby approved in partial fulfillment of the requirements for the award of B.Sc. Computer Science of the Department of Computer Science, Faculty of Computing. University of Benin, Benin City, Edo State, Nigeria to Akinirimisi Oladayo Efemena.

_____ _____

DR. (MRS.) USIOBAIFO A.R. **SIGNATURE/DATE**
(Head of Department)

UNDERTAKING

This project was carried out by **Akinirimisi Oladayo Efemena** with Matriculation number **PSC1814735**. I have not copied the work of any other author(s) All texts used have been duly cited and acknowledged

Akinirimisi Oladayo Efemena	**SIGNATURE/DATE**

DEDICATION

I dedicate this Project firstly to God, my family especially my beloved mother, for all her support, love and encouragement, my Lectures, Supervisors, Course mates and everyone who made this Project a success

ACKNOWLEDGEMENT

I express gratitude to the Almighty God for granting me the resilience to navigate through school amidst tough and demanding Circumstances' am extremely grateful and thankful to him for his grace, mercy, unconditional love, strength and kindness he has showed me throughout my study in the department of computer science.

I would like to express my sincere gratitude and appreciation to Dr. (Mrs) R.O Osaseri for her valuable guidance, support and mentorship throughout this project. And her expertise and encouragement were instrumental in shaping the direction of this work, May God bless you ma.

I am grateful and truly indebted to all my lecturers for their support, care, assistance, encouragement and guidance throughout my years of study, I am sincerely grateful.

I extend my heartfelt gratitude to my mum Mrs Evarister Esharive for bringing me to the world and for loving and protecting me in her own amazing way, I am grateful for the support, unconditional love and care she has shown me financially and spiritually, May God bless and keep you to enjoy the fruit of your labour, Amen.

ABSTRACT

The management of personnel records in many organizations remains largely manual, leading to inefficiencies, data redundancy, and delays in decision-making. This study focuses on the design and implementation of a computerized Personnel Management Information System (PMIS) to address these challenges and improve human resource management processes. The system was developed using Python as the programming language, MySQL as the database management system, and a web-based framework to ensure ease of access and usability.

The proposed system automates key human resource functions, including employee registration, attendance tracking, leave management, payroll computation, and report generation. A modular approach was adopted during implementation, ensuring each system component operates independently while maintaining seamless interaction with the central database. Testing was conducted using unit testing, integration testing, system testing, and user acceptance testing, demonstrating that the system is reliable, accurate, and efficient.

The results show that the developed system significantly improves personnel data management by reducing errors, enhancing data security, and providing timely reports for informed decision-making. This system offers a scalable solution that can be adopted by organizations seeking to modernize their human resource management operations, thereby demonstrating the importance of information technology in enhancing organizational efficiency.

TABLE OF CONTENT

CHAPTER ONE

INTRODUCTION

1.1 Background of the Study

In the contemporary business environment, effective personnel management has become a critical factor in organizational success and competitive advantage. Personnel management, also known as human resource management, encompasses the strategic and coherent approach to managing an organization's most valued assets the people working there who individually and collectively contribute to the achievement of organizational objectives (Armstrong and Taylor, 2020). The evolution of information technology has fundamentally transformed how organizations manage their human resources, shifting from traditional paper-based systems to sophisticated digital solutions that enhance efficiency, accuracy, and decision-making capabilities.

The digital transformation of human resource management has accelerated significantly in recent years, with organizations increasingly recognizing the need for integrated information systems to manage their workforce effectively. According to Obeidat (2020), the implementation of Human Resource Information Systems (HRIS) has become essential for organizations seeking to improve their HR service delivery, reduce administrative burden, and support strategic decision-making processes. A Personnel Management Information System (PMIS) serves as a comprehensive solution that automates and streamlines various HR functions including employee data management, attendance tracking, leave management, payroll processing, performance evaluation, and report generation (Kavanagh and Johnson, 2021).

The traditional manual approach to personnel management has proven increasingly inadequate in addressing the complexities of modern organizational structures. Manual systems are characterized

by numerous challenges including data redundancy, inconsistency in record keeping, time-consuming retrieval processes, limited accessibility, and vulnerability to human errors (Malik et al., 2021). Furthermore, the growing volume of employee data and the need for real-time information access have made computerized systems not merely an option but a necessity for organizations of all sizes (Boudreau and Cascio, 2020).

Recent studies have demonstrated that organizations implementing automated personnel management systems experience significant improvements in operational efficiency and strategic HR management. Research by Bondarouk et al. (2019) revealed that HRIS implementation leads to enhanced data accuracy, improved compliance with regulatory requirements, faster information processing, and better support for strategic human resource planning. Additionally, the COVID-19 pandemic has further accelerated the adoption of digital HR solutions, as organizations worldwide have been compelled to manage remote workforces and maintain business continuity through technology-enabled systems (Carnevale and Hatak, 2020).

The integration of database management systems with user-friendly interfaces has revolutionized personnel management practices, enabling HR professionals to focus on strategic initiatives rather than administrative tasks. According to Strohmeier and Parry (2020), modern PMIS solutions incorporate advanced features such as cloud computing, mobile accessibility, analytics capabilities, and artificial intelligence, which collectively enhance the value proposition of HR technology investments. These systems facilitate data-driven decision-making by providing managers and executives with timely, accurate, and comprehensive information about their workforce (Marler and Boudreau, 2021).

In developing economies and small to medium-sized enterprises, the adoption of personnel management information systems remains relatively low despite their proven benefits. This gap presents both a challenge and an opportunity for technological intervention. Studies by Nura and Osman (2019) indicate that many organizations in developing contexts continue to rely on manual or semi-automated systems due to factors such as cost constraints, limited technical expertise, resistance to change, and lack of awareness about available solutions. However, the development and implementation of customized, cost-effective PMIS solutions can address these barriers and enable organizations to leverage technology for improved personnel management outcomes.

The design and implementation of a personnel management information system requires careful consideration of organizational needs, technological capabilities, and user requirements. Contemporary system development approaches emphasize user-centered design, scalability, security, and integration capabilities to ensure that the resulting system meets both current and future organizational needs (Sommerville, 2020). The application of robust database design principles, secure authentication mechanisms, and intuitive user interfaces contributes to the development of effective PMIS solutions that deliver tangible value to organizations (Coronel and Morris, 2019).

This study focuses on the design and implementation of a personnel management information system that addresses the critical challenges faced by organizations in managing their human resources efficiently. By leveraging current technologies and best practices in system development, this project aims to create a comprehensive solution that automates key personnel management functions, enhances data integrity, improves accessibility, and supports informed decision-making processes.

1.2 Statement of the Problem

Despite the recognized importance of efficient personnel management in organizational success, many organizations continue to face significant challenges in managing their workforce information effectively. The persistence of manual and semi-automated personnel management systems has resulted in numerous operational inefficiencies that adversely affect organizational productivity and strategic HR management capabilities.

One of the primary problems confronting organizations is the inefficiency and time-consuming nature of manual record-keeping systems. According to research by Iqbal et al. (2019), manual personnel management processes consume excessive time and resources, with HR professionals spending up to 60% of their time on administrative tasks rather than strategic activities. The manual handling of employee records, leave applications, attendance tracking, and payroll calculations creates bottlenecks that slow down organizational processes and reduce overall productivity.

Data integrity and accuracy represent another critical challenge in traditional personnel management systems. Manual data entry and paper-based record-keeping are inherently prone to human errors, including transcription mistakes, duplicate entries, and inconsistent data formats (Sharma and Sharma, 2020). These errors can lead to serious consequences such as incorrect payroll calculations, compliance violations, and flawed decision-making based on inaccurate information. A study by Dessler (2020) found that organizations using manual HR systems experience error rates of up to 30% in their personnel records, significantly impacting operational efficiency and employee satisfaction.

The difficulty in retrieving and accessing employee information constitutes a major obstacle in manual systems. When personnel records are stored in physical files or disparate electronic

documents, locating specific information becomes a time-intensive process that frustrates both HR staff and employees (Noe et al., 2021). This challenge is particularly acute when managers require immediate access to employee data for decision-making purposes or when employees need to verify their records. The lack of centralized, easily accessible information repositories undermines the responsiveness and agility of HR functions.

Security and confidentiality of employee information pose significant concerns in traditional personnel management systems. Paper-based records and unsecured electronic files are vulnerable to unauthorized access, loss, damage, and theft (Mondy and Martocchio, 2020). With increasing regulatory requirements around data protection, including legislation such as the General Data Protection Regulation (GDPR) and various national data protection laws, organizations face legal and reputational risks when they cannot adequately secure sensitive employee information (Hoofnagle et al., 2019).

The generation of comprehensive reports and analytics represents another substantial challenge in manual systems. Organizations require various reports for decision-making, compliance, and strategic planning purposes, including headcount reports, turnover analysis, leave summaries, and payroll reports (Cascio and Boudreau, 2019). In manual systems, compiling such reports requires extensive manual effort, is prone to errors, and often results in outdated information by the time reports are completed. This limitation severely restricts the organization's ability to engage in data-driven HR management and strategic workforce planning.

Scalability and adaptability issues further compound the problems associated with traditional personnel management systems. As organizations grow and evolve, their personnel management needs become more complex, requiring systems that can accommodate increasing volumes of data

and expanding functionality (Stone et al., 2020). Manual systems lack the flexibility and scalability to adapt to changing organizational requirements, forcing organizations to continually modify their processes or invest in complete system overhauls.

Furthermore, the lack of integration between personnel management and other organizational systems creates information silos and duplication of effort. Modern organizations require seamless integration between HR, payroll, accounting, and other business functions to optimize operations and ensure data consistency (Boselie, 2019). The absence of such integration in manual or legacy systems results in redundant data entry, inconsistencies across systems, and missed opportunities for process optimization.

These challenges collectively undermine organizational efficiency, increase operational costs, compromise data quality, and limit the strategic contribution of human resource management functions. The need for a comprehensive, automated personnel management information system that addresses these multifaceted problems and enables organizations to manage their workforce effectively in the digital age forms the foundation for this research project.

1.3 Aim and Objectives of the Study

The aim of this study is to design and implement a comprehensive Personnel Management Information System (PMIS) that automates and streamlines human resource management processes, enhances data integrity and accessibility, and supports effective decision-making in organizations. While the objectives to the research is as follows:

1. **To conduct a comprehensive analysis of existing personnel management systems** and identify their inherent limitations, challenges, and areas requiring technological intervention. This objective involves examining current manual and semi-automated processes, documenting workflows, and understanding user requirements through systematic data collection methods.
2. **To design a robust and normalized database structure** that efficiently stores, organizes, and manages employee information while ensuring data integrity, minimizing redundancy, and supporting complex queries and reporting requirements. This includes developing entity-relationship models, defining table structures, establishing relationships, and implementing appropriate constraints and validation rules (Elmasri and Navathe, 2020).
3. **To develop an intuitive and user-friendly interface** that facilitates easy interaction with the system by users with varying levels of technical expertise. This objective emphasizes the principles of user-centered design, accessibility, and usability to ensure high user adoption rates and satisfaction (Preece et al., 2019).
4. **To implement core functional modules** that address essential personnel management requirements, including:
    - Employee registration and profile management module for capturing and maintaining comprehensive employee information

- Attendance tracking module for recording and monitoring employee attendance patterns
- Leave management module for processing leave applications, approvals, and tracking leave balances
- Payroll processing module for calculating salaries, deductions, and generating pay slips
- Report generation module for producing various management reports and analytics

5. **To incorporate robust security mechanisms** including user authentication, role-based access control, data encryption, and audit trails to protect sensitive employee information and ensure compliance with data protection regulations (Whitman and Mattord, 2021).

1.4 Significance of the Study

The design and implementation of a Personnel Management Information System carries substantial significance for multiple stakeholders and contributes to both practical organizational improvements and academic knowledge advancement in the field of information systems.

1.5 Scope and Limitations of the Study

This study encompasses the design and implementation of a Personnel Management Information System with clearly defined boundaries to ensure focused and achievable outcomes within the constraints of an undergraduate research project.

1.5.2 Limitations of the Study

Despite efforts to develop a comprehensive solution, this study acknowledges several limitations that bound the research scope and outcomes:

1. Advanced Features Exclusion:

The system does not incorporate certain advanced features found in sophisticated commercial HRIS solutions, such as:

- Talent acquisition and recruitment management modules
- Performance management and appraisal systems
- Training and development tracking
- Succession planning capabilities
- Compensation management and benefits administration
- Workforce analytics and predictive modeling using artificial intelligence (Bondarouk et al., 2019)

These exclusions are deliberate, focusing the project on core personnel management functions achievable within the timeframe and resources of an undergraduate project.

1.6 Definition of Terms

To ensure clarity and common understanding throughout this study, the following key terms are defined:

Personnel Management: The administrative function concerned with recruiting, selecting, developing, utilizing, compensating, and motivating employees to achieve organizational

objectives. Personnel management encompasses all activities related to managing people within an organization, including workforce planning, employee relations, and compliance with employment regulations (Armstrong and Taylor, 2020).

Management Information System (MIS): A computer-based system that provides managers and decision-makers with timely, accurate, and comprehensive information necessary for planning, organizing, directing, and controlling organizational activities. MIS transforms raw data into meaningful information through systematic processes of collection, storage, processing, and dissemination (Laudon and Laudon, 2020).

Personnel Management Information System (PMIS): An integrated computerized system specifically designed to manage all aspects of an organization's human resources, including employee data management, attendance tracking, leave management, payroll processing, and generation of personnel-related reports. PMIS serves as the technological backbone for modern human resource management practices (Kavanagh and Johnson, 2021).

Human Resource Information System (HRIS): A comprehensive software solution that combines human resource management activities with information technology to facilitate the planning, administration, development, and control of personnel functions. HRIS is often used interchangeably with PMIS, though HRIS typically encompasses broader strategic HR capabilities (Bondarouk et al., 2019).

Database: An organized collection of structured data stored electronically in a computer system, managed by a Database Management System (DBMS). A database enables efficient data storage,

retrieval, modification, and deletion while maintaining data integrity and supporting multiple users (Coronel and Morris, 2019).

Database Management System (DBMS): Software that provides an interface between users and databases, managing data storage, retrieval, security, and integrity. DBMS examples include MySQL, PostgreSQL, Oracle, and Microsoft SQL Server (Elmasri and Navathe, 2020).

User Interface (UI): The means by which users interact with a computer system, application, or website. UI includes all visual elements, controls, and interactive components that enable user-system communication. An effective UI is intuitive, efficient, and accessible (Preece et al., 2019).

Authentication: The process of verifying the identity of a user attempting to access a system, typically through credentials such as username and password. Authentication ensures that only authorized individuals can access system resources (Whitman and Mattord, 2021).

Authorization: The process of determining what actions or resources an authenticated user is permitted to access or manipulate within a system. Authorization is typically implemented through role-based access control mechanisms (Sommerville, 2020).

Module: A self-contained component of a software system that performs specific functions and can be developed, tested, and maintained independently while integrating with other modules. Modular design enhances system maintainability and scalability (Pressman and Maxim, 2020).

Payroll: The total amount of wages paid by an organization to its employees, as well as the process of calculating these wages, withholding appropriate deductions, and disbursing payments. Payroll

management includes maintaining records of earnings, deductions, and net pay for compliance and reporting purposes (Dessler, 2020).

Leave Management: The systematic process of tracking, approving, and managing employee absences from work, including vacation days, sick leave, personal days, and other types of authorized absences. Leave management ensures policy compliance and maintains adequate staffing levels (Noe et al., 2021).

System Development Life Cycle (SDLC): A structured process for planning, creating, testing, and deploying information systems. SDLC consists of distinct phases including requirements analysis, design, implementation, testing, deployment, and maintenance (Sommerville, 2020).

Requirements Analysis: The process of identifying, documenting, and validating the needs and expectations of system stakeholders. Requirements analysis forms the foundation for system design and implementation (Pressman and Maxim, 2020).

Data Integrity: The accuracy, consistency, and reliability of data throughout its lifecycle. Data integrity ensures that information remains correct and unaltered except through authorized processes, maintaining its trustworthiness for decision-making (Elmasri and Navathe, 2020).

Normalization: The process of organizing data in a database to reduce redundancy and improve data integrity by dividing large tables into smaller, related tables and defining relationships between them. Normalization follows specific rules or normal forms (Coronel and Morris, 2019).

Encryption: The process of converting data into a coded format to prevent unauthorized access. Encryption protects sensitive information during storage and transmission, ensuring confidentiality and security (Whitman and Mattord, 2021).

Scalability: The capability of a system to handle growing amounts of work or to be readily enlarged to accommodate growth. A scalable system can expand its capacity by adding resources without requiring fundamental architectural changes (Sommerville, 2020).

Stakeholder: Any individual, group, or organization that has an interest in or is affected by a system. In personnel management systems, stakeholders include HR staff, employees, managers, executives, and IT personnel (Pressman and Maxim, 2020).

Digital Transformation: The integration of digital technology into all areas of organizational operations, fundamentally changing how organizations operate and deliver value. In HR context, digital transformation involves adopting technology-enabled processes to enhance workforce management (Strohmeier and Parry, 2020).

These definitions provide the conceptual framework for understanding the technical and functional aspects discussed throughout this study and ensure consistent interpretation of key concepts.

CHAPTER TWO

LITERATURE REVIEW

2.1 INTRODUCTION

This chapter presents a comprehensive review of relevant literature on personnel management information systems, examining theoretical frameworks, empirical studies, and technological developments that inform this research. The review is structured to provide a thorough understanding of personnel management concepts, information systems theory, existing solutions, technological components, and system development methodologies. By synthesizing current knowledge from academic and professional sources, this chapter establishes the theoretical foundation for the design and implementation of the proposed Personnel Management Information System (PMIS).

The literature review examines scholarly articles, books, conference proceedings, and industry reports published primarily between 2019 and 2024, ensuring that the research is grounded in contemporary knowledge and reflects current trends in human resource information systems. This temporal focus is particularly important given the rapid technological advancements and digital transformation initiatives that have characterized recent years, especially following the COVID-19 pandemic which accelerated organizational adoption of digital HR solutions (Carnevale and Hatak, 2020).

The chapter is organized into thematic sections that progressively build understanding from conceptual foundations through technical implementations. This structure facilitates a logical progression from theoretical concepts to practical applications, ultimately identifying gaps in existing literature that justify the current research project.

2.2 Conceptual Framework

2.2.1 Personnel Management: Concepts and Evolution

Personnel management, contemporarily referred to as human resource management (HRM), represents a critical organizational function concerned with managing people to achieve strategic objectives. According to Armstrong and Taylor (2020), personnel management encompasses all activities related to attracting, developing, motivating, and retaining employees who contribute to organizational success. The evolution of personnel management from administrative personnel departments to strategic human capital management reflects broader organizational transformations and the recognition of human resources as sources of competitive advantage.

The traditional view of personnel management focused primarily on administrative functions such as record-keeping, payroll processing, and compliance with labor regulations (Dessler, 2020). However, contemporary perspectives emphasize strategic contributions, including talent acquisition, organizational development, performance management, and workforce planning. This evolution has been accompanied by increasing reliance on technology to manage both administrative and strategic HR functions effectively (Noe et al., 2021).

Boselie (2019) articulates that modern personnel management operates at three levels: strategic (aligning HR practices with organizational strategy), managerial (implementing policies and managing employee relations), and operational (executing day-to-day HR transactions). Information systems support all three levels by providing data-driven insights for strategic decisions, automating managerial processes, and streamlining operational activities. The integration of technology with personnel management has given rise to what scholars term "digital

HRM" or "e-HRM," representing the planning and implementation of information technology for networking and supporting HR functions (Strohmeier and Parry, 2020).

Research by Malik et al. (2021) introduces the concept of "augmented human resource management," wherein artificial intelligence, machine learning, and advanced analytics enhance traditional HR functions. While such sophisticated capabilities may exceed the scope of basic PMIS implementations, they represent the trajectory of personnel management evolution and underscore the importance of developing systems with scalable architectures that can accommodate future enhancements.

2.2.2 Management Information Systems in Organizational Context

Management Information Systems (MIS) constitute integrated systems that provide information to support organizational decision-making, coordination, control, analysis, and visualization (Laudon and Laudon, 2020). The fundamental purpose of MIS is to transform raw data into meaningful information through systematic processes of collection, storage, processing, and dissemination. In the personnel management context, MIS applications specifically address the information needs of HR professionals, managers, and employees regarding workforce-related matters.

The architecture of MIS typically comprises five components: hardware, software, data, procedures, and people (Oz and Jones, 2020). Hardware includes physical devices such as servers, computers, and network infrastructure. Software encompasses operating systems, database management systems, and application programs. Data represents the facts and figures stored and processed by the system. Procedures define the policies and rules governing system operation. People include all individuals who interact with the system, from developers to end-users.

Stair and Reynolds (2020) identify several characteristics of effective management information systems: flexibility to adapt to changing organizational needs, timeliness in providing information when needed, accuracy in representing reality correctly, completeness in providing all necessary information, relevance by focusing on information that matters for decisions, verifiability through traceable data sources, accessibility to authorized users, and security to protect sensitive information from unauthorized access. These characteristics serve as design criteria for developing personnel management information systems.

The value proposition of MIS in organizations extends beyond operational efficiency to strategic enablement. According to research by Mardiana et al. (2020), organizations implementing effective information systems experience improvements in decision-making quality, operational efficiency, customer satisfaction, competitive positioning, and financial performance. In the personnel management domain, these benefits manifest as enhanced workforce planning, improved employee experiences, better compliance management, and more strategic HR contributions to organizational success.

2.2.3 Integration of Information Technology in Human Resource Management

The integration of information technology with human resource management has fundamentally transformed HR practice over recent decades. Bondarouk et al. (2019) conducted a comprehensive review spanning four decades of research on electronic HRM (e-HRM) adoption and consequences, identifying three primary goals for e-HRM implementation: improving HR service efficiency, improving HR strategic orientation, and improving client service. Their analysis revealed that while efficiency improvements are consistently achieved, strategic benefits remain more elusive and depend heavily on implementation approaches and organizational contexts.

Obeidat (2020) investigated the impact of Human Resource Information Systems on strategic HRM in Jordanian commercial banks, finding that HRIS implementation significantly enhances strategic HR management capabilities including workforce planning, talent management, and organizational development. The study demonstrated that technology adoption enables HR professionals to shift focus from transactional activities to transformational leadership, contributing more substantially to organizational strategy formulation and execution.

The concept of "HR analytics" or "people analytics" represents a significant dimension of IT integration in HRM. Marler and Boudreau (2021) define HR analytics as a methodology for creating insights on organizational performance by integrating HR data with business data and applying statistical methods. Their evidence-based review highlights how analytics capabilities embedded in HRIS enable predictive modeling, workforce segmentation, and evidence-based decision-making. However, they note that many organizations struggle to progress beyond descriptive analytics to more sophisticated predictive and prescriptive analytics, often due to data quality issues, skill gaps, and organizational culture constraints.

Research by Iqbal et al. (2019) examining e-HRM in Pakistani commercial banks demonstrated that e-HRM implementation improves labor productivity through enhanced information access, streamlined processes, and better communication. Their findings indicated that employee self-service capabilities, in particular, contribute significantly to productivity improvements by empowering employees to manage their own HR transactions and reducing administrative burden on HR staff.

The COVID-19 pandemic has accelerated digital transformation in HRM, with organizations rapidly adopting cloud-based HRIS, virtual recruitment tools, remote performance management

systems, and digital learning platforms (Carnevale and Hatak, 2020). This acceleration has highlighted both the potential and challenges of digital HRM, including concerns about employee surveillance, digital divides, and the need for human-centered technology design that supports rather than constrains employee autonomy and well-being.

2.3 Theoretical Framework

2.3.1 Systems Theory

Systems theory provides a foundational framework for understanding organizational information systems. Originating from biology and subsequently applied across multiple disciplines, systems theory conceptualizes organizations as complex systems comprising interrelated components that interact to achieve common objectives (Mele et al., 2020). In the context of personnel management information systems, systems theory emphasizes the interconnectedness of various HR functions and the need for integrated solutions that address these relationships holistically.

According to systems theory, a system is characterized by inputs, processes, outputs, and feedback mechanisms (Skyttner, 2020). For a PMIS, inputs include employee data, attendance records, leave applications, and policy parameters. Processes encompass data validation, calculations, workflow routing, and report generation. Outputs consist of employee records, pay slips, reports, and management information. Feedback mechanisms involve user interactions, system performance monitoring, and continuous improvement initiatives.

The concept of system boundaries is particularly relevant for PMIS design. Boundaries define what is included within the system and what constitutes the external environment (Rousseau, 2019). Clear boundary definition ensures focused system development while recognizing interfaces with external systems such as accounting software, time tracking devices, or government reporting

systems. The principle of equifinality, which suggests that systems can achieve the same end state through different pathways, supports flexible design approaches that accommodate organizational variations while achieving common objectives.

2.3.2 Information Processing Theory

Information Processing Theory (IPT) explains how organizations collect, store, manipulate, and distribute information to support decision-making and coordination (Tushman and Nadler, 2019). IPT posits that organizations are information processing systems that must develop capacities matching the information processing requirements imposed by their tasks and environments. For personnel management, IPT suggests that as organizations grow and HR functions become more complex, the need for sophisticated information processing capabilities increases proportionally.

Galbraith's information processing framework identifies two fundamental strategies for managing increased information processing requirements: reducing the need for information processing or increasing information processing capacity (Colombo et al., 2019). PMIS implementation represents the latter strategy, increasing organizational capacity to handle complex HR information through automation, integration, and analytical capabilities. The framework also emphasizes the importance of fit between information processing requirements and capabilities, suggesting that systems must be designed to match organizational complexity and environmental demands.

Research by Bharadwaj et al. (2020) extends information processing theory to examine how digital technologies enhance organizational information processing capabilities. Their findings indicate that cloud computing, mobile technologies, and analytics platforms collectively enable organizations to process information faster, more accurately, and with greater accessibility than

traditional systems. These enhanced capabilities support more agile and responsive personnel management practices.

2.3.3 Human Capital Theory

Human Capital Theory (HCT) provides an economic perspective on personnel management, treating investments in people as analogous to investments in physical capital (Becker, 2020). HCT posits that investments in employee development, health, and well-being generate returns through enhanced productivity, innovation, and organizational performance. This theoretical lens justifies investments in personnel management systems as mechanisms for optimizing human capital utilization and development.

Wright et al. (2021) apply human capital theory to strategic HRM, arguing that effective human capital management requires accurate information about employee capabilities, performance, and development needs. PMIS serves as the infrastructure enabling organizations to track human capital investments, measure returns, and make informed decisions about workforce development. The theory emphasizes that human capital is context-specific and relationship-dependent, suggesting that personnel systems must capture not only individual employee characteristics but also team dynamics, organizational culture, and contextual factors affecting performance.

Contemporary extensions of human capital theory recognize intellectual capital as comprising human capital, structural capital, and relational capital (Mention and Bontis, 2019). This tripartite conceptualization suggests that personnel management systems should extend beyond individual employee management to capture organizational knowledge, processes, and relationships. While such comprehensive coverage may exceed basic PMIS scope, the theoretical framework provides direction for system evolution and enhancement.

2.3.4 Technology Acceptance Model

The Technology Acceptance Model (TAM), originally proposed by Davis (1989) and extensively refined by subsequent researchers, explains user acceptance and usage of information systems (Marangunić and Granić, 2019). TAM posits that perceived usefulness and perceived ease of use are primary determinants of technology acceptance, mediated by attitudes and behavioral intentions. Understanding TAM is crucial for PMIS development because system success depends not only on technical capabilities but also on user acceptance and adoption.

Recent TAM extensions incorporate additional variables including subjective norms, facilitating conditions, and system quality (Venkatesh et al., 2020). The Unified Theory of Acceptance and Use of Technology (UTAUT) integrates multiple theoretical perspectives, identifying performance expectancy, effort expectancy, social influence, and facilitating conditions as key determinants of technology acceptance. For PMIS implementation, these theories suggest that systems must be designed with careful attention to usability, must deliver clear value to users, and must be supported by adequate training and technical infrastructure.

Research by Alam et al. (2020) examining HRIS adoption in developing countries found that perceived usefulness significantly influences adoption intentions, while perceived ease of use indirectly affects adoption through its impact on perceived usefulness. Their findings emphasize the importance of user-centered design that prioritizes intuitive interfaces and clearly communicates system benefits to potential users. Additionally, they identified organizational support and top management commitment as critical contextual factors moderating technology acceptance.

2.3.5 Resource-Based View

The Resource-Based View (RBV) of the firm provides a strategic perspective on organizational capabilities and competitive advantage (Barney and Hesterly, 2019). RBV argues that sustained competitive advantage derives from valuable, rare, inimitable, and non-substitutable (VRIN) resources and capabilities. Human resources and the systems managing them can constitute strategic resources when they meet VRIN criteria.

Wright and McMahan (2020) apply RBV to strategic HRM, arguing that while individual employees may not be inimitable, the collective human capital pool, organizational culture, and HR systems that develop and deploy talent can constitute sources of competitive advantage. PMIS contributes to competitive advantage by enabling more effective talent identification, development, and retention while supporting data-driven workforce strategies that competitors cannot easily replicate.

The dynamic capabilities perspective extends RBV by emphasizing organizational abilities to integrate, build, and reconfigure internal and external competencies to address rapidly changing environments (Teece, 2019). For personnel management, dynamic capabilities include the ability to rapidly redeploy workforce, develop new competencies, and adapt HR practices to changing business needs. Information systems support dynamic capabilities by providing real-time workforce visibility, facilitating agile talent management, and enabling scenario planning.

2.4 Review of Existing Personnel Management Systems

2.4.1 Commercial HRIS Solutions

The market for Human Resource Information Systems has experienced substantial growth, with numerous vendors offering solutions ranging from comprehensive enterprise resource planning (ERP) systems with HR modules to specialized HRIS platforms. Understanding existing

commercial solutions provides valuable insights for PMIS development and helps identify features, architectures, and design patterns that have proven effective in practice.

SAP SuccessFactors represents one of the leading cloud-based HRIS platforms, offering comprehensive functionality including core HR, talent management, workforce analytics, and employee experience management (SAP, 2021). According to Johnson et al. (2020), SAP SuccessFactors' strength lies in its integration capabilities, analytical tools, and scalability, making it suitable for large multinational corporations. However, its complexity and cost make it inaccessible for many small and medium-sized enterprises, creating opportunities for simpler, more affordable alternatives.

Workday HCM provides another prominent cloud-based solution emphasizing unified data architecture, mobile accessibility, and continuous innovation (Workday, 2021). Research by Gupta and Seetharaman (2020) highlights Workday's intuitive user interface and robust reporting capabilities as key differentiators. The platform employs object-oriented architecture that treats all entities (employees, positions, organizations) as objects with defined attributes and relationships, providing flexibility for complex organizational structures.

Oracle HCM Cloud offers comprehensive HR functionality integrated with Oracle's broader enterprise application suite (Oracle, 2021). The platform leverages artificial intelligence and machine learning for recruiting, talent management, and predictive analytics. However, Kavanagh and Johnson (2021) note that Oracle's solutions tend to appeal primarily to large organizations with existing Oracle infrastructure, limiting adoption among smaller entities.

BambooHR targets small to medium-sized businesses with a simplified HRIS emphasizing ease of use and core HR functions (BambooHR, 2021). Unlike enterprise solutions, BambooHR focuses on essential features such as employee data management, time-off tracking, and basic reporting, making it more accessible for organizations with limited IT resources. Research by Stone et al. (2020) identifies BambooHR as exemplifying the trend toward user-friendly, focused applications that prioritize usability over comprehensive functionality.

Zoho People provides an affordable, cloud-based HRIS with modular architecture allowing organizations to implement specific functions based on their needs (Zoho, 2021). The platform emphasizes customization, workflow automation, and integration with other Zoho applications. According to Nura and Osman (2019), Zoho People's pricing model and scalability make it particularly attractive for organizations in developing economies seeking to digitize HR functions incrementally.

2.4.2 Open-Source HRIS Solutions

Open-source HRIS solutions offer alternatives to commercial systems, providing cost advantages and customization flexibility. These systems have gained traction among organizations with technical capabilities to implement and maintain them.

OrangeHRM stands as one of the most widely adopted open-source HRIS platforms, offering core HR, personnel information management, leave management, time and attendance, and recruitment modules (OrangeHRM, 2021). Research by Dessler (2020) indicates that OrangeHRM's modular architecture allows organizations to implement specific modules based on their requirements while maintaining the option to expand functionality over time. The platform's

active community contributes plugins, translations, and support, enhancing its viability for diverse organizational contexts.

Odoo provides an open-source ERP system with comprehensive HR modules including employee management, attendance, leave, expenses, recruitment, and appraisal (Odoo, 2021). According to Coronel and Morris (2019), Odoo's integrated approach connecting HR with accounting, project management, and other business functions represents a strength for organizations seeking unified business management solutions. However, the breadth of functionality may introduce complexity for organizations primarily seeking HR capabilities.

IceHRM offers a simpler open-source alternative focused specifically on core HR functions including employee management, leave tracking, and basic reporting (IceHRM, 2021). The platform emphasizes ease of installation and use, making it accessible for small organizations with limited technical resources. However, its limited feature set may not satisfy complex organizational requirements.

2.4.3 Academic and Research Projects

Numerous academic research projects have explored HRIS development, providing valuable insights into design approaches, implementation challenges, and evaluation methodologies.

Adebayo (2019) developed a web-based personnel management system for a Nigerian university, employing PHP and MySQL to create modules for employee records, leave management, and report generation. The study demonstrated the feasibility of developing customized HRIS solutions for specific organizational contexts while highlighting challenges including user resistance, data migration from legacy systems, and the need for continuous training and support.

Research by Munirat et al. (2020) presented the design and implementation of an automated human resource management system using Laravel framework and MySQL database. Their system incorporated biometric attendance integration, providing a practical example of HRIS extension with hardware devices. The study's evaluation revealed significant improvements in attendance tracking accuracy and reduction in time theft, demonstrating tangible benefits of automation.

Oladele and Adeoti (2021) developed a personnel information management system emphasizing security features including encryption, access control, and audit trails. Their research highlighted the importance of robust security implementations in personnel systems given the sensitive nature of employee data and increasing regulatory requirements around data protection. The study provided detailed security architecture that can inform similar projects.

Khan et al. (2020) explored cloud-based HRIS development for small businesses, emphasizing accessibility, scalability, and cost-effectiveness. Their solution utilized cloud infrastructure (Amazon Web Services) to provide SaaS-style access to HR functionality, demonstrating how cloud computing can democratize access to sophisticated HRIS capabilities for resource-constrained organizations. The study documented deployment considerations including data residency requirements, internet connectivity dependencies, and subscription pricing models.

2.4.4 Comparative Analysis and Gap Identification

Comparative analysis of existing systems reveals several patterns and identifies gaps that justify the current research project. Commercial systems, while feature-rich and reliable, remain inaccessible for many organizations due to cost, complexity, and implementation requirements (Boselie, 2019). Typical annual costs for commercial HRIS range from $5 to $15 per employee

per month for basic systems to $35+ for comprehensive platforms, creating substantial financial barriers for small organizations with limited budgets (Kavanagh and Johnson, 2021).

Open-source solutions offer cost advantages but present challenges related to technical expertise requirements, limited documentation, and support availability (Stone et al., 2020). Organizations lacking IT departments or technical staff may struggle to implement, customize, and maintain open-source systems effectively, potentially resulting in abandoned implementations or suboptimal utilization.

Academic projects demonstrate the feasibility of developing context-specific HRIS solutions but often remain proof-of-concept implementations without progressing to production-ready systems (Sharma and Sharma, 2020). Additionally, documentation of these projects sometimes lacks sufficient detail for replication or adaptation by other researchers or practitioners.

Gap analysis reveals several opportunities for the current research:

1. **Simplicity versus Functionality Balance**: Existing systems tend toward either comprehensive complexity (enterprise solutions) or limited functionality (simplified systems). There is opportunity for solutions that balance essential functionality with ease of use and implementation.
2. **Cost-Effectiveness**: While open-source solutions exist, many require substantial technical expertise for implementation and customization. There is need for well-documented, easily implementable solutions suitable for organizations with limited technical resources.
3. **Contextual Adaptation**: Most commercial systems reflect Western organizational structures and HR practices, potentially misaligning with organizations in different cultural

or regulatory contexts (Nura and Osman, 2019). Customizable solutions adaptable to local requirements address this gap.

4. **Educational Resources**: Limited academic projects provide comprehensive documentation suitable for educational purposes or replication by other researchers. Detailed documentation of system development processes contributes to knowledge dissemination.
5. **Integration Readiness**: While comprehensive integration capabilities characterize enterprise systems, smaller solutions often lack consideration for future integration needs. Architectures designed with integration in mind from inception facilitate future expansion.

These identified gaps inform the design and implementation approach for the current project, emphasizing practical functionality, clear documentation, cost-effectiveness, and contextual relevance.

2.5 Database Management Systems for Personnel Information

2.5.1 Database Concepts and Importance

Databases constitute the foundation of modern information systems, providing structured mechanisms for storing, organizing, and retrieving data efficiently. According to Elmasri and Navathe (2020), a database is a collection of related data representing some aspect of the real world, designed for specific purposes and intended for specific user groups. Database Management Systems (DBMS) are software systems that provide interfaces for defining, constructing, manipulating, and sharing databases among users and applications.

The importance of proper database design for personnel management systems cannot be overstated. Employee data is characterized by complexity (multiple related entities), longevity (records maintained over decades), sensitivity (requiring robust security), and regulatory significance (subject to various compliance requirements). Coronel and Morris (2019) emphasize that well-designed databases ensure data integrity, minimize redundancy, support efficient queries, facilitate maintenance, and scale gracefully as organizational needs evolve.

2.5.2 Relational Database Model

The relational database model, introduced by E.F. Codd and now dominant in database implementations, represents data as tables (relations) consisting of rows (tuples) and columns (attributes) (Date, 2019). Relationships between tables are established through common attributes, typically implemented using primary and foreign keys. The relational model's mathematical foundation provides theoretical guarantees about data consistency and supports powerful query languages like SQL (Structured Query Language).

For personnel management applications, the relational model offers several advantages. First, it naturally represents organizational structures and relationships such as employees belonging to departments, reporting to supervisors, and holding positions (Elmasri and Navathe, 2020). Second, relational databases support ACID properties (Atomicity, Consistency, Isolation, Durability) ensuring transaction integrity crucial for payroll calculations and other sensitive operations (Coronel and Morris, 2019). Third, mature relational DBMS platforms provide extensive tooling, documentation, and community support facilitating development and maintenance.

2.5.3 Database Normalization

Normalization represents a systematic approach to organizing database structures to minimize redundancy and dependency anomalies. The process involves applying a series of normal forms, each addressing specific types of data redundancy and update anomalies (Date, 2019). The primary normal forms include:

First Normal Form (1NF) requires that each attribute contain only atomic values and that each row be unique. For personnel databases, this means avoiding repeating groups such as storing multiple phone numbers in a single field (Elmasri and Navathe, 2020).

Second Normal Form (2NF) eliminates partial dependencies, requiring that non-key attributes depend on the entire primary key. This is relevant when using composite primary keys, ensuring attributes relate to the complete key rather than portions thereof (Coronel and Morris, 2019).

Third Normal Form (3NF) removes transitive dependencies, requiring that non-key attributes depend directly on the primary key rather than on other non-key attributes. For example, storing

department location with each employee record would violate 3NF if location is determined by department; instead, location should reside in a separate department table (Date, 2019).

Higher normal forms (BCNF, 4NF, 5NF) address more subtle anomalies and dependencies. While these forms ensure maximum data integrity, practical implementations often accept denormalization for performance optimization in specific scenarios (Elmasri and Navathe, 2020). Research by Coronel and Morris (2019) suggests that achieving 3NF provides an appropriate balance between data integrity and practical implementation complexity for most business applications including PMIS.

2.5.4 Popular DBMS Platforms

Several database management systems are commonly employed for personnel management applications, each with distinct characteristics, advantages, and limitations.

MySQL represents the most widely used open-source relational database system, offering reliability, ease of use, and strong community support (Dubois, 2020). According to Sharma and Sharma (2020), MySQL's advantages include zero licensing cost, extensive documentation, cross-platform compatibility, and good performance for read-heavy workloads typical in HRIS applications. Its ACID compliance, foreign key support, and transaction capabilities make it suitable for personnel management systems requiring data integrity. However, MySQL historically exhibited limitations in handling complex queries and large-scale concurrent writes, though recent versions have addressed many such concerns.

PostgreSQL provides another popular open-source DBMS, emphasizing standards compliance, extensibility, and advanced features (Smith, 2020). PostgreSQL supports sophisticated data types,

complex queries, full ACID compliance, and advanced features like table inheritance and custom functions. Research by Coronel and Morris (2019) identifies PostgreSQL as particularly suitable for applications requiring complex data models and sophisticated queries. For personnel management systems incorporating advanced analytics or complex organizational hierarchies, PostgreSQL offers significant advantages. However, its configuration and optimization may require more expertise than MySQL.

Microsoft SQL Server offers a comprehensive commercial DBMS widely adopted in Windows-based enterprise environments (Microsoft, 2021). SQL Server provides robust security features, excellent integration with Microsoft technologies, powerful business intelligence tools, and comprehensive management utilities. According to Kavanagh and Johnson (2021), SQL Server's advantages for HRIS include enterprise-grade security, high availability features, and seamless integration with .NET applications. However, licensing costs and Windows dependency limit its applicability in resource-constrained or cross-platform scenarios.

Oracle Database represents the leading enterprise-level DBMS, offering extensive features, scalability, and reliability (Oracle, 2021). Oracle excels in handling large-scale, mission-critical applications with complex requirements. However, its high cost and complexity make it less suitable for small to medium-sized PMIS implementations (Elmasri and Navathe, 2020).

SQLite provides a lightweight, serverless database engine embedded within applications (SQLite, 2021). While unsuitable for multi-user HRIS implementations, SQLite serves well for mobile applications, offline capabilities, or single-user scenarios. Its simplicity and zero configuration requirements offer advantages for specific use cases within broader personnel management ecosystems.

2.5.5 Database Design for Personnel Management

Effective database design for personnel management systems requires careful entity identification, relationship modeling, and attribute definition. Core entities typically include:

Employee Entity: Central to the system, storing personal information, contact details, employment dates, and status. According to Armstrong and Taylor (2020), employee entities should accommodate diverse employment types including full-time, part-time, contract, and temporary workers.

Department Entity: Represents organizational units, storing department names, descriptions, locations, and hierarchical relationships. Coronel and Morris (2019) emphasize that department structures should support reorganizations without data integrity compromises.

Position Entity: Defines roles within the organization, including job titles, descriptions, requirements, and salary ranges. Noe et al. (2021) advocate separating position definitions from employee assignments to maintain clarity when positions are vacant or multiple employees hold the same position.

Attendance Entity: Records employee time-related information including clock-in/out times, work hours, and attendance status. Research by Dessler (2020) suggests designing attendance structures to accommodate various organizational policies including flexible schedules and remote work arrangements.

Leave Entity: Manages leave applications, approvals, and balances for various leave types. Kavanagh and Johnson (2021) recommend flexible leave type definitions allowing organizations to configure different leave policies based on employment terms, tenure, or local regulations.

Payroll Entity: Stores salary components, deductions, and payment history. Given its sensitivity and regulatory importance, payroll database design requires particular attention to security, audit trails, and calculation accuracy (Mondy and Martocchio, 2020).

Relationships between these entities reflect organizational realities: employees belong to departments, hold positions, record attendance, request leaves, and receive payments. Proper relationship definition using foreign keys ensures referential integrity, preventing orphaned records and maintaining data consistency (Elmasri and Navathe, 2020).

2.6 Programming Technologies and Development Tools

2.6.1 Web-Based Application Development

Contemporary PMIS implementations increasingly favor web-based architectures offering advantages including cross-platform compatibility, centralized deployment, simplified maintenance, and accessibility from diverse devices (Sommerville, 2020). Web-based systems eliminate the need for client software installation, reduce IT support burdens, and facilitate remote access increasingly important in hybrid work environments (Carnevale and Hatak, 2020).

Web application architecture typically follows a multi-tier pattern separating presentation logic (user interface), application logic (business rules), and data management (database). This separation enhances maintainability, scalability, and security by establishing clear boundaries between system components (Pressman and Maxim, 2020). Modern web architectures increasingly employ microservices patterns, breaking applications into smaller, independently deployable services, though such sophistication may exceed requirements for basic PMIS implementations.

2.6.2 Server-Side Programming Languages

Several programming languages dominate server-side web development, each with distinct characteristics suitable for PMIS implementation.

Python has gained significant traction in web development due to its readability, extensive libraries, and versatility (McKinney, 2020). Frameworks like Django and Flask provide robust foundations for web application development. Django, in particular, emphasizes "batteries included" philosophy, offering built-in features for authentication, database ORM (Object-Relational Mapping), form handling, and security (Forcier et al., 2020). Research by Malik et al. (2021) identifies Python's strong data analysis capabilities as advantageous for HRIS applications incorporating analytics. However, Python's performance may lag behind compiled languages in compute-intensive scenarios.

PHP remains widely used for web development, particularly in small to medium-sized applications (Tatroe et al., 2020). PHP's advantages include extensive hosting availability, large ecosystem of frameworks and libraries, and gentle learning curve. Laravel, a modern PHP framework, provides elegant syntax, robust security features, and comprehensive tooling for rapid application development (Otwell, 2021). According to Stone et al. (2020), PHP's maturity and widespread adoption make it a pragmatic choice for PMIS development, though its inconsistent standard library and historical security vulnerabilities require careful attention.

JavaScript (Node.js) enables server-side JavaScript execution, allowing full-stack JavaScript development using the same language for client and server (Hahn, 2020). Node.js excels in handling concurrent connections and real-time communications, making it suitable for applications requiring live updates or chat functionality. However, its asynchronous programming model introduces complexity that may challenge developers without JavaScript expertise.

Java provides a mature, enterprise-grade platform for web application development (Sierra and Bates, 2020). Frameworks like Spring Boot offer comprehensive features for building robust, scalable applications. Java's strong typing, extensive tooling, and mature ecosystem make it attractive for large-scale implementations requiring high reliability. However, its verbosity and complexity may introduce development overhead for smaller projects (Sommerville, 2020).

2.6.3 Client-Side Technologies

Client-side technologies determine user interface behavior, interactivity, and user experience.

HTML5 provides the structural foundation for web pages, defining content organization and semantics (Robbins, 2020). HTML5 introduced semantic elements, form enhancements, and multimedia support improving web application capabilities. Proper HTML5 usage enhances accessibility, search engine optimization, and maintainability.

CSS3 controls visual presentation including layouts, colors, fonts, and responsive design (Meyer, 2021). Modern CSS frameworks like Bootstrap, Tailwind CSS, and Materialize CSS provide pre-designed components and responsive grid systems accelerating interface development while ensuring cross-device compatibility (Preece et al., 2019). Research by Lazar et al. (2020) emphasizes that responsive design is now essential given the diversity of devices accessing web applications.

JavaScript enables interactive client-side functionality including form validation, dynamic content updates, and responsive interfaces (Haverbeke, 2020). Modern JavaScript frameworks and libraries like React, Vue.js, and Angular facilitate sophisticated single-page applications (SPAs) offering desktop-like user experiences within web browsers (Banks and Porcello, 2020). However,

SPA complexity may exceed requirements for straightforward PMIS implementations where traditional server-rendered pages suffice.

2.6.4 Development Frameworks

Frameworks provide structured approaches to application development, offering reusable components, design patterns, and best practices that accelerate development while improving code quality.

Django (Python) emphasizes rapid development and pragmatic design, incorporating an ORM, authentication system, admin interface, and security features (Forcier et al., 2020). Its "don't repeat yourself" (DRY) principle and "convention over configuration" approach reduce boilerplate code and decision fatigue. Django's admin interface automatically generates CRUD (Create, Read, Update, Delete) interfaces for database models, providing immediate functionality for data management.

Laravel (PHP) offers elegant syntax, expressive ORM (Eloquent), robust routing, and comprehensive authentication and authorization systems (Otwell, 2021). Laravel's ecosystem includes tools for queue management, task scheduling, and testing. Its active community and extensive documentation facilitate learning and problem-solving.

Spring Boot (Java) simplifies Java web application development through convention-based configuration and embedded servers (Walls, 2020). Spring Boot's comprehensive feature set supports enterprise requirements including security, data access, cloud deployment, and microservices architecture. However, its complexity and learning curve may challenge developers new to the ecosystem.

Framework selection should consider factors including developer expertise, performance requirements, community support, documentation quality, and long-term maintainability (Sommerville, 2020). For educational projects like the current study, frameworks with gentle learning curves and comprehensive documentation may be preferable to maximize learning outcomes and minimize implementation obstacles.

2.6.5 Integrated Development Environments

Integrated Development Environments (IDEs) and code editors significantly impact developer productivity by providing syntax highlighting, code completion, debugging tools, and project management capabilities (Spinellis, 2020). Popular options include:

Visual Studio Code: A lightweight, extensible editor supporting multiple languages through extensions (Microsoft, 2021). Its large ecosystem, git integration, and debugging capabilities make it highly versatile for web development.

PyCharm: JetBrains' Python-focused IDE offering intelligent code completion, advanced debugging, and Django support (JetBrains, 2021). PyCharm's refactoring tools and database integration benefit Python-based PMIS development.

NetBeans/Eclipse: Mature IDEs particularly strong for Java development, offering comprehensive project management and debugging capabilities (Apache, 2021).

Tool selection often reflects personal preference and language focus, with modern editors increasingly supporting multiple languages and frameworks through extensions.

2.7 System Development Methodologies

2.7.1 Waterfall Model

The Waterfall model represents the traditional sequential approach to software development, progressing linearly through phases: requirements analysis, design, implementation, testing, deployment, and maintenance (Sommerville, 2020). Each phase must complete before the next begins, with formal reviews validating completion criteria.

Advantages of the Waterfall model include clear structure, comprehensive documentation, and well-defined milestones facilitating project management (Pressman and Maxim, 2020). The model suits projects with stable, well-understood requirements where changes are unlikely. For academic projects with defined scopes and timelines, Waterfall's structure provides clarity and predictability.

However, the Waterfall model's rigidity constitutes a significant limitation. Once a phase completes, returning to previous phases to accommodate requirement changes proves difficult and expensive (Sommerville, 2020). Additionally, working software emerges only late in the development process, delaying feedback and increasing risk of building inappropriate solutions. Research by Balaji and Murugaiyan (2020) suggests Waterfall remains viable for small projects with clear requirements but recommends iterative approaches for complex or evolving requirements.

2.7.2 Agile Methodology

Agile methodologies emerged as alternatives to Waterfall, emphasizing iterative development, continuous feedback, and adaptive planning (Beck et al., 2001; Pressman and Maxim, 2020). Agile approaches deliver working software incrementally, incorporating user feedback throughout development rather than waiting until completion.

Scrum, a popular Agile framework, organizes work into time-boxed sprints (typically 2-4 weeks) delivering potentially shippable increments (Schwaber and Sutherland, 2020). Scrum ceremonies including daily standups, sprint planning, sprint reviews, and retrospectives facilitate communication and continuous improvement. For PMIS development, Scrum enables incremental module delivery, allowing users to provide feedback on completed modules while subsequent modules remain in development.

Extreme Programming (XP) emphasizes technical excellence through practices including pair programming, test-driven development, continuous integration, and refactoring (Beck,

2020). XP practices improve code quality and reduce defects, though they require discipline and may increase initial development time.

Agile advantages include flexibility to accommodate changing requirements, early and continuous delivery of value, and strong stakeholder engagement (Sommerville, 2020). However, Agile's success depends heavily on client availability for regular feedback and team self-organization capabilities. Research by Campanelli and Parreiras (2020) indicates that Agile adoption in academic contexts faces challenges including time constraints, limited stakeholder availability, and assessment structures favoring comprehensive upfront documentation.

2.7.3 Prototyping Model

The Prototyping model emphasizes building working versions of the system early in development to clarify requirements and validate design decisions (Pressman and Maxim, 2020). Prototypes may be throwaway (discarded after requirements clarification) or evolutionary (refined into the final system).

For PMIS development, prototyping offers significant advantages. HR professionals may struggle to articulate detailed requirements without seeing working examples. Prototypes provide concrete artifacts facilitating requirement discussions and design validation (Sommerville, 2020). User interface prototypes, in particular, help stakeholders visualize system appearance and behavior, leading to more accurate requirements and better user satisfaction.

However, prototyping risks include stakeholder expectations that prototypes represent complete systems, potential for prototypes evolving into production systems without proper engineering, and time spent on prototypes potentially delaying actual development (Pressman and Maxim, 2020). Mitigating these risks requires clear communication about prototype purposes and limitations, along with disciplined decisions about when to discard or refine prototypes.

2.7.4 Spiral Model

The Spiral model combines iterative development with systematic risk assessment, proceeding through repeated cycles of planning, risk analysis, engineering, and evaluation (Boehm, 2020). Each spiral iteration produces increasingly refined versions while explicitly addressing project risks.

The Spiral model's risk-focused approach suits projects with significant uncertainties or potential for costly failures (Sommerville, 2020). For PMIS development, major risks might include data security breaches, regulatory non-compliance, or user rejection. The Spiral model's explicit risk assessment helps ensure adequate attention to such concerns.

However, the Spiral model's complexity and overhead make it potentially excessive for small projects with limited risks. Risk assessment activities require expertise and time that may not be justified for straightforward implementations (Pressman and Maxim, 2020). Research by Balaji and Murugaiyan (2020) suggests reserving Spiral model applications for large, complex, or high-risk projects rather than routine development efforts.

2.7.5 Methodology Selection for PMIS Development

Selecting an appropriate development methodology requires balancing project characteristics, team capabilities, stakeholder availability, and organizational context. For undergraduate PMIS projects, several factors influence methodology choice:

Project Timeline: Academic projects operate within fixed semester or academic year timelines, favoring methodologies with clear milestones and predictable progression (Pressman and Maxim, 2020).

Requirement Stability: If stakeholders provide clear, stable requirements upfront, Waterfall's structured approach may suffice. However, if requirements evolve through stakeholder interaction, iterative approaches prove more suitable (Sommerville, 2020).

Assessment Requirements: Academic assessment often requires comprehensive documentation demonstrating systematic approaches. Methodologies emphasizing documentation may better satisfy assessment criteria (Campanelli and Parreiras, 2020).

Team Experience: Less experienced developers may benefit from structured methodologies providing clear guidance, while experienced teams may prefer Agile's flexibility (Balaji and Murugaiyan, 2020).

For the current study, a modified Waterfall approach incorporating prototyping elements provides an appropriate balance. The clear phase structure facilitates project management and documentation while prototyping activities validate designs before full implementation. This hybrid approach accommodates academic requirements while mitigating Waterfall's limitation regarding late feedback.

2.8 System Security Considerations

Security constitutes a critical concern for personnel management systems given the sensitive nature of employee data and regulatory requirements around data protection. Comprehensive security implementations address multiple dimensions including authentication, authorization, data protection, and audit.

2.8.1 Authentication and Access Control

Authentication verifies user identity, typically through credentials like usernames and passwords, before granting system access (Whitman and Mattord, 2021). Strong authentication practices include:

Password Policies: Enforcing minimum password complexity, regular password changes, and preventing password reuse enhances security (Furnell, 2020). However, overly burdensome policies may lead users to adopt insecure workarounds like writing passwords down or using predictable patterns.

Multi-Factor Authentication (MFA): Requiring multiple authentication factors (something you know, something you have, something you are) significantly increases security by making account compromise more difficult even if passwords are stolen (Grassi et al., 2020). While MFA implementation may exceed basic PMIS scope, architecting systems to accommodate future MFA addition demonstrates security consciousness.

Session Management: Properly managing user sessions prevents unauthorized access through stolen session tokens. Best practices include using secure, HTTP-only cookies, implementing session timeouts, and invalidating sessions upon logout (OWASP, 2021).

Authorization determines what authenticated users can access and modify. **Role-Based Access Control (RBAC)** assigns permissions to roles rather than individual users, simplifying administration and ensuring consistent policy enforcement (Sandhu et al., 2020). For PMIS, typical roles might include:

- **Administrator**: Full system access including user management and system configuration
- **HR Manager**: Access to all employee data, ability to approve leaves and process payroll
- **Department Manager**: Access to their department's employee data, ability to approve subordinate leaves
- **Employee**: Access to personal data, ability to request leaves and view pay slips

Implementing fine-grained authorization prevents privilege escalation and data exposure (Whitman and Mattord, 2021).

2.8.2 Data Encryption

Encryption protects sensitive data from unauthorized access both at rest (stored data) and in transit (transmitted data).

Transport Layer Security (TLS): Implementing HTTPS encrypts data transmitted between users and servers, preventing interception and eavesdropping (Rescorla, 2020). Modern web applications should exclusively use HTTPS, with HTTP traffic redirected to secure channels. Let's Encrypt and similar services provide free TLS certificates removing cost barriers to encryption.

Database Encryption: Encrypting sensitive fields like social security numbers, bank account details, and salary information protects data even if database backups or files are compromised (Elmasri and Navathe, 2020). However, encryption impacts query performance and complicates searching, requiring careful consideration of which fields warrant encryption.

Password Hashing: Storing passwords as salted hashes rather than plaintext protects credentials even if databases are breached (Grassi et al., 2020). Modern hashing algorithms like bcrypt, scrypt, or Argon2 resist brute-force attacks through computational intensity. Never storing plaintext passwords constitutes a fundamental security principle.

2.8.3 Input Validation and SQL Injection Prevention

SQL Injection attacks exploit inadequate input validation to inject malicious SQL code, potentially exposing or modifying database contents (OWASP, 2021). Preventing SQL injection requires:

Parameterized Queries: Using prepared statements with parameter binding ensures user input is treated as data rather than executable code (Coronel and Morris, 2019). Modern frameworks typically provide ORM layers that automatically generate parameterized queries, reducing injection risks.

Input Validation: Validating and sanitizing all user input ensures conformance to expected formats, rejecting potentially malicious content (Whitman and Mattord, 2021). Validation should occur server-side even if client-side validation exists, as client-side controls can be bypassed.

Least Privilege Database Access: Database accounts used by applications should possess minimum necessary permissions, preventing attackers from exploiting application vulnerabilities to perform administrative database operations (Elmasri and Navathe, 2020).

2.8.4 Cross-Site Scripting (XSS) Prevention

XSS attacks inject malicious scripts into web pages viewed by other users, potentially stealing session cookies, redirecting users to malicious sites, or performing unauthorized actions (OWASP, 2021). Prevention strategies include:

Output Encoding: Encoding user-generated content before displaying prevents browsers from interpreting content as executable code (Stuttard and Pinto, 2019). Context-appropriate encoding (HTML, JavaScript, URL) ensures proper protection.

Content Security Policy (CSP): CSP headers instruct browsers to restrict resource loading, preventing inline script execution and limiting script sources to trusted domains (Weissbacher et al., 2020).

2.8.5 Audit Trails and Logging

Comprehensive logging records system activities, supporting security monitoring, compliance demonstration, and incident investigation (Whitman and Mattord, 2021). Effective audit trails log:

- Authentication events (successful and failed logins)
- Authorization changes (role assignments, permission modifications)
- Data access and modifications (who accessed or changed what data, when)
- Administrative actions (system configuration changes)
- Security events (suspected attacks, policy violations)

Log protection through restricted access and integrity verification prevents tampering that might conceal malicious activities (Furnell, 2020). However, logging must balance security with privacy, avoiding excessive collection of sensitive personal information.

2.9 Summary and Gap Identification

This literature review has examined personnel management concepts, theoretical frameworks, existing systems, technological components, development methodologies, and security considerations relevant to PMIS development. The review reveals several key insights:

1. **Digital Transformation Imperative**: Organizations increasingly recognize PMIS as essential for efficient HR management, strategic workforce planning, and competitive advantage (Strohmeier and Parry, 2020).

2. **System Diversity**: Available solutions range from expensive enterprise platforms to open-source alternatives, each with distinct tradeoffs between functionality, cost, complexity, and support (Kavanagh and Johnson, 2021).
3. **Technology Maturity**: Mature, accessible technologies exist for developing robust PMIS implementations, including capable databases, expressive programming languages, and comprehensive frameworks (Coronel and Morris, 2019; Sommerville, 2020).
4. **Security Criticality**: Given sensitive employee data and regulatory requirements, security must be fundamental rather than afterthought in PMIS design (Whitman and Mattord, 2021).
5. **Methodology Flexibility**: Multiple valid development approaches exist, with methodology selection depending on project characteristics, team capabilities, and stakeholder availability (Pressman and Maxim, 2020).

Despite substantial existing knowledge, gaps remain that justify the current research

CHAPTER THREE

SYSTEM ANALYSIS AND DESIGN

3.1 INTRODUCTION

This chapter focuses on the analysis and design of the proposed Personnel Management Information System (PMIS). It presents a detailed examination of the existing personnel management processes within the selected organization and outlines the systematic approach adopted in designing a computerized solution. System analysis is a critical phase in information system development, as it involves understanding the current system, identifying its weaknesses, and determining user and organizational requirements that the proposed system must satisfy.

According to Laudon and Laudon (2020), effective system analysis ensures that the developed system aligns with organizational goals and addresses real operational challenges rather than perceived problems. In the context of this study, the analysis phase examines the manual personnel management practices used by the case study organization in Benin City, highlighting issues such as inefficiency, data redundancy, slow information retrieval, and poor record security.

The design aspect of this chapter translates the identified requirements into a structured system blueprint. This includes the architectural design, database design, process modeling, interface design, and security considerations of the proposed PMIS. Proper system design ensures that the final implementation is scalable, user-friendly, secure, and capable of supporting core personnel management functions such as employee record management, leave administration, payroll processing, and report generation (Dennis, Wixom, & Roth, 2021).

3.2 System Development Methodology

A System Development Methodology (SDM) refers to a structured framework used to plan, analyze, design, implement, and evaluate an information system. Selecting an appropriate methodology is essential because it determines how system requirements are gathered, how development activities are sequenced, and how risks are managed throughout the project lifecycle (Sommerville, 2020).

For this study, the **Waterfall Model** was adopted as the system development methodology.

Overview of the Waterfall Model

The Waterfall Model is a sequential and linear approach to system development in which each phase must be completed before the next begins. The typical phases include:

1. Requirements Analysis
2. System Design
3. Implementation
4. Testing
5. Deployment
6. Maintenance

Each phase produces deliverables that serve as inputs to the subsequent phase, ensuring proper documentation and control throughout the development process (Pressman & Maxim, 2020).

Justification for the Chosen Methodology

The Waterfall Model was selected for this project for several reasons:

1. **Clearly Defined Requirements**

   The personnel management processes of the case study organization are well-structured and stable, involving standard activities such as employee registration, leave management, payroll processing, and report generation. Since system requirements are clearly understood at the outset, a linear methodology like the Waterfall Model is appropriate (Dennis et al., 2021).

2. **Suitability for Academic Projects**

   The Waterfall Model is widely accepted in academic environments due to its simplicity, clarity, and emphasis on documentation. Nigerian universities and polytechnics often recommend this methodology for undergraduate projects because it allows easy supervision, assessment, and presentation of each development phase (Adebayo & Yusuf, 2019).

3. **Emphasis on Documentation**

This project requires extensive documentation, including requirement specifications, system design diagrams, test cases, and user manuals. The Waterfall Model supports thorough documentation at every stage, making it ideal for research-oriented system development (Sommerville, 2020).

4. **Reduced Development Complexity**

   Unlike Agile or iterative methodologies, which require frequent user feedback and continuous changes, the Waterfall Model minimizes complexity by following a predetermined sequence. This is suitable for a small-scale system developed within a limited academic timeframe.

5. **Ease of Evaluation and Testing**

   The structured nature of the Waterfall Model allows each module of the PMIS to be tested systematically after implementation. Errors can be traced back to specific development phases, improving system reliability and validation accuracy (Pressman & Maxim, 2020).

In summary, the Waterfall Model provides a disciplined, systematic, and well-documented approach that aligns with the objectives of this study. Its adoption ensures that the Personnel Management Information System is developed in a controlled manner, resulting in a reliable and efficient system that meets organizational needs.

Gap 1 - Accessibility: Commercial solutions remain cost-prohibitive for many organizations, particularly SMEs in developing economies, while open-source alternatives require technical expertise that may not be available (Nura and Osman, 2019). Need exists for well-documented, readily implementable solutions.

Gap 2 - Educational Resources: Limited academic projects provide comprehensive documentation suitable for replication or adaptation by other researchers and practitioners. Detailed documentation of design decisions, implementation approaches, and lessons learned contributes to knowledge dissemination.

Gap 3 - Contextual Relevance: Most documented PMIS implementations reflect specific organizational or geographic contexts. Additional case studies from diverse settings enhance understanding of contextual factors affecting system requirements and implementation success.

Gap 4 - Integration Architecture: While enterprise systems emphasize integration, many smaller implementations treat integration as afterthought, creating difficulties when organizations later require connectivity with other systems. Architectures designed from inception to accommodate future integration needs address this gap.

Gap 5 - User Experience Focus: Technical literature emphasizes functionality and security but sometimes neglects user experience and usability. Designing systems with explicit attention to user-centered design principles addresses this oversight (Preece et al., 2019).

The current research addresses these gaps by developing a well-documented, cost-effective PMIS emphasizing essential functionality, robust security, user-friendly design, and integration readiness. The following chapters detail the system analysis, design, and implementation, contributing practical and educational value to the personnel management information systems domain.

3.3 Analysis of the Existing System

This section examines the current manual personnel management system used by the case study organization in Benin City. The analysis focuses on how personnel records are presently handled, the challenges inherent in the system, and the flow of information within the manual process. Understanding the existing system is essential for identifying weaknesses and defining requirements for the proposed computerized Personnel Management Information System (PMIS).

3.3.1 Description of the Current Manual Personnel Management System

The existing personnel management system in the organization operates primarily through manual, paper-based procedures. Employee information such as personal details, employment history, attendance records, leave applications, and payroll data are recorded using physical files, registers, and forms. These records are stored in filing cabinets within the human resources office.

When a new employee is recruited, personal and employment details are captured manually on paper forms and filed under designated folders. Attendance is recorded using sign-in sheets or registers, while leave requests are processed through handwritten application forms that require multiple approvals. Payroll calculations are carried out manually using attendance records and salary schedules, after which payment summaries are prepared and forwarded for approval.

Retrieval of personnel information requires physical searching through files, which is often time-consuming. Report generation, such as staff lists, leave summaries, or payroll reports, involves manual compilation and calculation, increasing the likelihood of errors. The system depends heavily on human effort, making it inefficient in handling large volumes of personnel data as the organization grows.

3.3.2 Problems Identified in the Existing System

The analysis of the current manual system revealed several operational challenges, including the following:

1. **Data Redundancy and Inconsistency**

   Employee information is often duplicated across multiple files and registers, leading to inconsistencies and conflicting records.

2. **Slow Information Retrieval**

   Searching for employee records manually consumes significant time, especially when files are misplaced or poorly organized.

3. **High Risk of Human Error**

   Manual data entry, calculations, and record updates increase the chances of errors in attendance tracking, leave balances, and payroll computation.

4. **Poor Record Security**

   Physical files are vulnerable to unauthorized access, loss, fire, or damage, with limited mechanisms for data protection and confidentiality.

5. **Inefficient Report Generation**

   Management reports are difficult to generate promptly, as they require manual compilation and verification of data from multiple sources.

6. **Limited Decision Support**

   The absence of real-time data and analytical tools makes it difficult for management to make timely and informed personnel-related decisions.

7. **Scalability Challenges**

   As the number of employees increases, the manual system becomes increasingly difficult to manage and maintain effectively.

These problems significantly reduce operational efficiency and highlight the need for an automated personnel management solution.

3.3.3 Information Flow in the Manual Personnel Management Process

The flow of information in the existing manual system follows a linear and document-dependent pattern:

1. **Data Collection**

   Employee data is collected through handwritten forms during recruitment or updates.

2. **Record Storage**

   Completed forms are filed manually in cabinets under categorized folders.

3. **Processing**

   Attendance registers and leave forms are manually reviewed to compute work hours, leave balances, and payroll figures.

4. **Approval**

   Processed documents are forwarded physically to supervisors or management for authorization.

5. **Output Generation**

   Reports such as payroll summaries or staff lists are prepared manually and submitted to management.

6. **Archiving**

   Used documents are stored for future reference, increasing file volume over time.

This manual information flow lacks automation, integration, and real-time processing, resulting in delays, inaccuracies, and inefficiencies. These limitations form the basis for the design of the proposed computerized Personnel Management Information System described in subsequent sections.

3.4 Analysis of the Proposed System

This section presents an analysis of the proposed computerized Personnel Management Information System (PMIS). It describes the overall structure of the system, outlines the functional and non-functional requirements, and specifies user requirements. The proposed system is designed to address the limitations identified in the existing manual system by automating personnel management processes, improving data accuracy, enhancing security, and supporting efficient decision-making.

3.4.1 Summary of the New Computerized System

The proposed Personnel Management Information System is a computer-based application designed to automate and centralize personnel-related operations within the organization. The system will replace manual, paper-based processes with a digital platform that allows authorized users to capture, store, process, and retrieve employee information efficiently.

The system will maintain a centralized database containing employee biodata, employment history, attendance records, leave details, payroll information, and administrative reports. It will provide role-based access, ensuring that only authorized personnel can view or modify sensitive information. Routine tasks such as employee registration, leave processing, payroll calculation, and report generation will be automated, thereby reducing human errors and operational delays.

The PMIS is expected to improve data integrity, enhance record security, support real-time information access, and provide management with accurate reports needed for effective personnel planning and decision-making.

3.4.2 Functional Requirements

Functional requirements define what the system should do. The proposed system shall be able to perform the following functions:

1. **User Authentication and Authorization**
    - Allow users to log in using unique usernames and passwords.
    - Restrict system access based on user roles (e.g., administrator, HR officer, management).
2. **Employee Records Management**
    - Register new employees and capture biodata and employment details.
    - Update and delete employee records when necessary.
    - Search and retrieve employee information efficiently.
3. **Attendance Management**
    - Record employee attendance data.
    - View and generate attendance summaries.
    -
4. **Leave Management**
    - Allow employees to apply for leave.
    - Enable supervisors or HR officers to approve or reject leave requests.
    - Automatically update leave balances.
5. **Payroll Processing**
    - Calculate employee salaries based on predefined rules.
    - Generate payroll summaries and payslips.

6. Report Generation

- Generate reports such as staff lists, attendance reports, leave reports, and payroll reports.
- Allow reports to be viewed or printed.

7. Data Backup and Recovery

1. Support periodic data backup to prevent data loss.

3.4.3 Non-Functional Requirements

Non-functional requirements describe how the system should perform and define quality attributes of the system. These include:

1. **Performance**
   - The system should respond to user requests within an acceptable time frame.
   - Data retrieval and report generation should be fast and efficient.
2. **Security**
   - User authentication mechanisms must prevent unauthorized access.
   - Sensitive personnel data must be protected against data breaches and unauthorized modification.
3. **Usability**
   - The system interface should be user-friendly and easy to navigate.
   - Minimal training should be required for users to operate the system.
4. **Reliability**
   - The system should function correctly with minimal downtime.
   - Errors should be handled gracefully without data corruption.

5. **Scalability**
    - The system should support future growth in the number of employees and data volume.
6. **Maintainability**
    - The system should be easy to update and modify when organizational policies change.

3.4.4 User Requirement Specifications

User requirement specifications define the needs of different categories of system users. The proposed system identifies the following user groups:

1. **System Administrator**
    - Manage user accounts and access rights.
    - Perform system maintenance and backups.
2. **Human Resource Officer**
    - Register and update employee records.
    - Manage attendance, leave, and payroll information.
    - Generate personnel-related reports.
3. **Management**
    - View summary reports for decision-making.
    - Access high-level personnel statistics.
4. **Employees**
    - View personal information.
    - Apply for leave and check leave status.
    - View payroll details where permitted.

The proposed PMIS is designed to meet the needs of all identified user groups while ensuring data security, operational efficiency, and system reliability.

3.5 System Design

System design translates the requirements identified during system analysis into a **technical blueprint** that guides system implementation. This phase focuses on defining the system architecture, database structure, process models, user interfaces, and security mechanisms. A well-designed system ensures that the developed application is efficient, scalable, secure, and capable of meeting organizational needs.

For this study, the system design emphasizes simplicity, reliability, and ease of use, considering the operational environment of medium-sized organizations in Benin City. The proposed design aims to replace the inefficient manual system with a structured computerized solution that supports core personnel management functions while allowing for future system enhancements.

3.5.1 Architectural Design

The proposed Personnel Management Information System adopts a web-based client–server architecture. This architectural model separates system responsibilities into distinct layers, enabling efficient processing, centralized data management, and ease of access for authorized users.

System Structure

The architecture consists of three main components:

1. **Client Layer (Presentation Layer)**

The client layer represents the user interface through which users interact with the system. It is accessed via a web browser on desktop or mobile devices within the organization. This layer provides interfaces for login, employee registration, leave application, payroll viewing, and report generation. The use of a browser-based interface eliminates the need for specialized software installation on client machines.

2. **Application Server (Business Logic Layer)**

The application server hosts the core system logic and controls system operations. It processes user requests received from the client layer, enforces business rules, validates data, and manages system workflows such as leave approval and payroll computation. This layer also handles user authentication and authorization to ensure secure access to system resources.

3. **Database Server (Data Layer)**

The database server stores all personnel-related data, including employee records, attendance logs, leave information, payroll data, and system user accounts. A relational database management system (RDBMS) is used to ensure data consistency, integrity, and efficient retrieval. Centralized data storage enhances security and simplifies backup and recovery processes.

Justification for the Chosen Architecture

The web-based client–server architecture was chosen for the following reasons:

- **Accessibility:** Authorized users can access the system from different locations within the organization using standard web browsers.
- **Centralized Data Management:** Personnel data is stored in a single database, reducing data redundancy and inconsistency.

- **Scalability:** The architecture supports future expansion, such as adding new modules or increasing the number of users.
- **Ease of Maintenance:** System updates and maintenance are performed on the server without requiring changes on client devices.
- **Cost Effectiveness:** The use of open-source web technologies reduces implementation and maintenance costs.

Overall, the selected architecture provides a flexible and efficient platform for implementing the Personnel Management Information System, ensuring reliable performance and secure handling of sensitive personnel data.

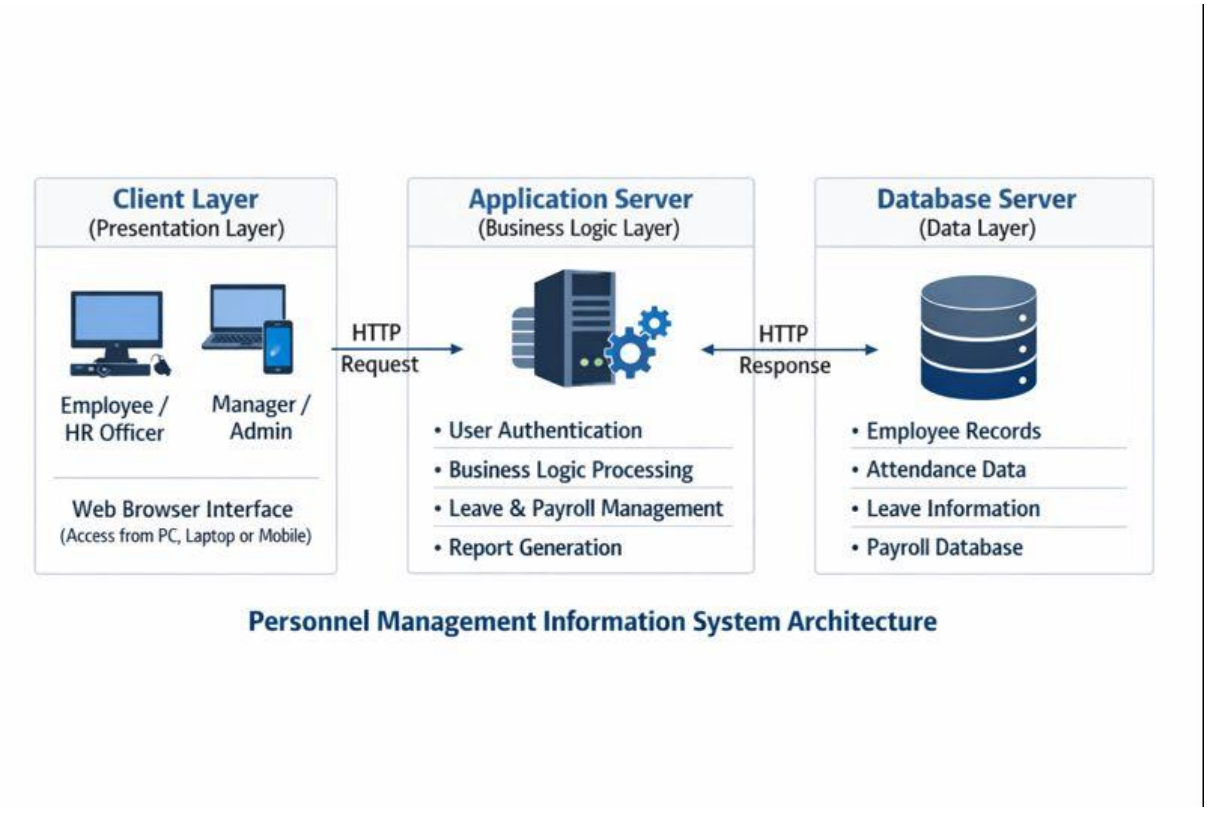

Fig 3.1: personnel management information system Architecture design

3.5.2 Database Design

Database design is a critical component of the Personnel Management Information System, as it determines how personnel data is stored, organized, and accessed. A well-structured database ensures data integrity, minimizes redundancy, and supports efficient data retrieval and reporting. For this study, a **relational database design approach** was adopted because it is suitable for handling structured personnel data and supports data consistency through defined relationships.

The database design process involved the development of **Entity–Relationship (ER) diagrams,** definition of table structures, and application of **normalization principles** to ensure efficient data organization.

Entity–Relationship (ER) Diagram

The Entity–Relationship diagram provides a graphical representation of the database structure, showing the key entities, their attributes, and the relationships among them. The main entities identified in the Personnel Management Information System include:

- Employee
- Department
- Attendance
- Leave
- Payroll
- User Account

Key Entities and Relationships

- An **Employee** belongs to one **Department**, while a Department can have many Employees (one-to-many relationship).
- An **Employee** can have multiple **Attendance** records.
- An **Employee** can submit multiple **Leave** applications.
- Each **Employee** is associated with one or more **Payroll** records.
- A **User Account** is linked to an Employee for system access and authentication.

These relationships ensure proper linkage of personnel data and support efficient querying and reporting.

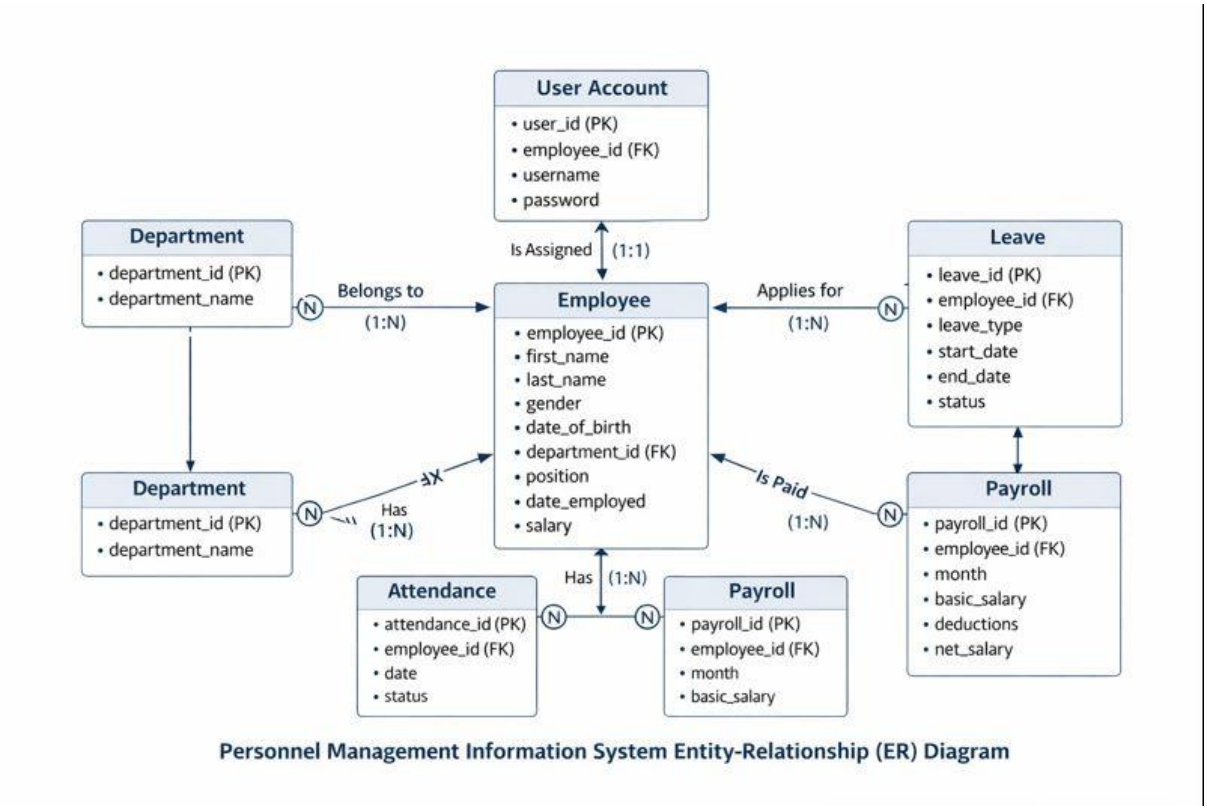

Fig 3.2: personnel Management information system entity relationship diagram

Table Structures

Based on the ER diagram, the following tables were designed:

Employee Table

Field Name	Data Type	Description
employee_id	INT (PK)	Unique employee identifier
first_name	VARCHAR	Employee first name
last_name	VARCHAR	Employee last name
Gender	VARCHAR	Gender
date_of_birth	DATE	Date of birth
department_id	INT (FK)	Department identifier
position	VARCHAR	Job title
date_employed	DATE	Employment start date
Salary	DECIMAL	Basic salary

Department Table

Field Name	Data Type	Description

department_id	INT (PK)	Unique department identifier
department_name	VARCHAR	Name of department

Attendance Table

Field Name	Data Type	Description
attendance_id	INT (PK)	Attendance record ID
employee_id	INT (FK)	Employee identifier
date	DATE	Attendance date
status	VARCHAR	Present/Absent

Leave Table

Field Name	Data Type	Description
leave_id	INT (PK)	Leave record ID
employee_id	INT (FK)	Employee identifier
leave_type	VARCHAR	Type of leave
start_date	DATE	Leave start date
end_date	DATE	Leave end date

status	VARCHAR	Approved/Pending/Rejected

Payroll Table

Field Name	Data Type	Description
payroll_id	INT (PK)	Payroll record ID
employee_id	INT (FK)	Employee identifier
Month	VARCHAR	Payroll month
basic_salary	DECIMAL	Basic salary
deductions	DECIMAL	Salary deductions
net_salary	DECIMAL	Net salary

User Table

Field Name	Data Type	Description
user_id	INT (PK)	User account ID
employee_id	INT (FK)	Linked employee
username	VARCHAR	Login username
password	VARCHAR	Encrypted password
role	VARCHAR	User role

Normalization

Normalization is the process of organizing data to reduce redundancy and improve data integrity. The database for this system was normalized up to the **Third Normal Form (3NF)**.

First Normal Form (1NF)

- All tables contain atomic (indivisible) values.
- Each record is uniquely identifiable using a primary key.

Second Normal Form (2NF)

- All non-key attributes depend fully on the primary key.
- Partial dependencies were eliminated by separating employee and department data into distinct tables.

Third Normal Form (3NF)

- Transitive dependencies were removed.
- Non-key attributes depend only on the primary key (e.g., department name depends on department_id, not employee_id).

Applying normalization ensures:

- Reduced data redundancy
- Improved data consistency
- Easier database maintenance
- Efficient query performance

Summary

The database design for the proposed Personnel Management Information System ensures efficient storage, secure management, and easy retrieval of personnel data. The use of ER diagrams, well-defined table structures, and normalization principles provides a solid foundation for implementing a reliable and scalable system.

3.5.3 Process Design

Process design involves modeling how data flows through the system and how users interact with the system. This section uses **Use Case Diagrams** to represent user interactions and **Data Flow Diagrams (DFDs)** to illustrate the flow of data within the proposed Personnel Management Information System (PMIS).

A. Use Cases

Use cases describe the different ways users interact with the system to achieve specific goals. The PMIS has the following primary actors:

Actors

1. **System Administrator**
2. **HR Officer**
3. **Manager**
4. **Employee**

Use Case Descriptions

Use Case 1: User Authentication

- **Actor:** System Administrator, HR Officer, Manager, Employee
- **Description:** Users log in to the system using valid credentials. The system verifies credentials and grants access based on user roles.
- **Pre-condition:** User must be registered in the system.
- **Post-condition:** User is redirected to their dashboard.

Use Case 2: Employee Registration

- **Actor:** HR Officer
- **Description:** HR officer registers new employees by capturing biodata and employment details.
- **Pre-condition:** HR officer must be authenticated.
- **Post-condition:** New employee record is stored in the database.

Use Case 3: Attendance Recording

- **Actor:** HR Officer
- **Description:** HR officer records daily attendance for employees.
- **Pre-condition:** HR officer is authenticated.
- **Post-condition:** Attendance records are updated in the database.

Use Case 4: Leave Application

- **Actor:** Employee
- **Description:** Employee applies for leave using the system.

- **Pre-condition:** Employee must be logged in.
- **Post-condition:** Leave request is submitted for approval.

Use Case 5: Leave Approval

- **Actor:** Manager / HR Officer
- **Description:** Manager/HR reviews and approves or rejects leave applications.
- **Pre-condition:** Leave application exists in the system.
- **Post-condition:** Leave status is updated.

Use Case 6: Payroll Processing

- **Actor:** HR Officer
- **Description:** System calculates payroll based on attendance, leave status, and salary data.
- **Pre-condition:** Attendance and leave records must be updated.
- **Post-condition:** Payroll records are generated.

Use Case 7: Report Generation

- **Actor:** Manager / HR Officer
- **Description:** Users generate reports such as employee list, attendance summary, leave report, and payroll report.
- **Pre-condition:** User must be authenticated.
- **Post-condition:** Report is generated and available for printing or download.

Use Case Diagram (Description)

The Use Case Diagram for the PMIS includes actors (Admin, HR Officer, Manager, Employee) and their interactions with system functions such as login, employee management, attendance, leave, payroll, and reporting.

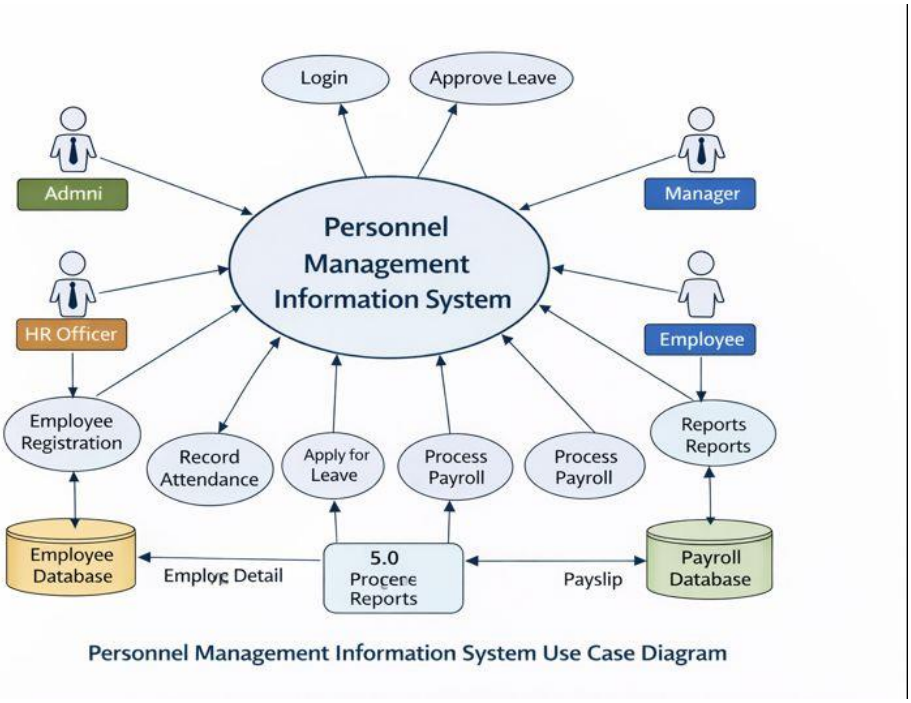

Fig 3.3: personnel management information system use case diagram

B. Data Flow Diagrams (DFDs)

DFDs show how data moves through the system and the processes that transform data. The proposed PMIS is modeled using:

- **Context Diagram (Level 0 DFD)**
- **Level 1 DFD**

DFD Context Diagram (Level 0)

External Entities

- Employees

- HR Officer
- Manager

Main System

- Personnel Management Information System (PMIS)

Data Flows

- Employees submit leave requests and view payslips.
- HR officer submits attendance data and manages employee records.
- Manager reviews reports and approves leave.

Level 1 DFD

Level 1 DFD breaks down the main system into sub-processes:

1. **User Authentication**
    - Validates user credentials
    - Grants access based on role
2. **Employee Management**
    - Add/Update/Delete employee records
    - Store employee details
3. **Attendance Management**
    - Record daily attendance
    - Update attendance database
4. **Leave Management**
    - Receive leave application
    - Approve/Reject leave

- Update leave records

5. **Payroll Processing**
    - Calculate salary based on attendance and leave
    - Generate payslips
6. **Report Generation**
    - Produce attendance, leave, and payroll reports

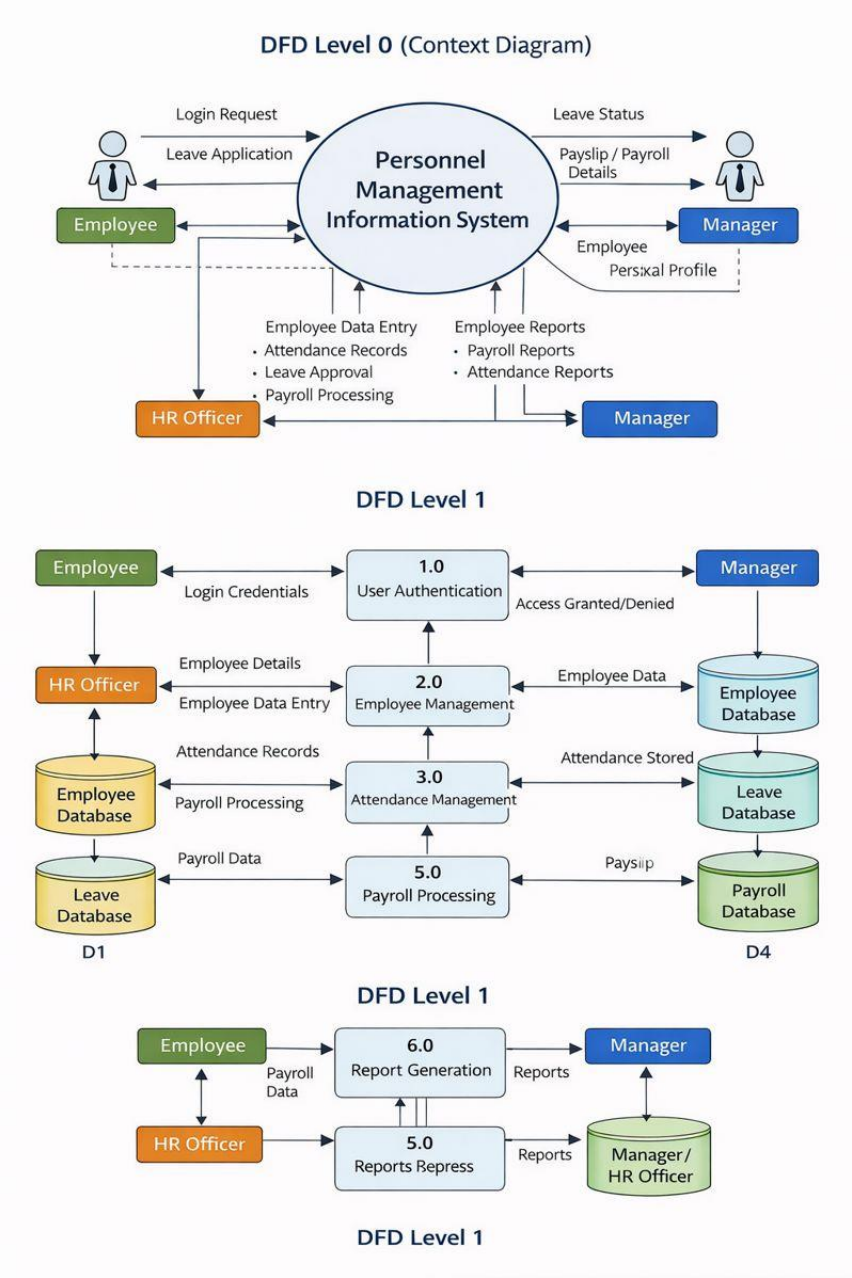

Fig 3.4: Dfd level 0, and Level 1

CHAPTER FOUR

DATA PRESENTATION AND ANALYSIS

4.1 INTRODUCTION

This chapter presents the implementation and testing of the proposed Personnel Management Information System (PMIS). Following the system analysis and design discussed in Chapter Three, this chapter focuses on the practical realization of the system, detailing how the designed components were translated into a functional and operational application.

The chapter explains the technologies and tools adopted for system development, the implementation of the database and functional modules, and the security measures incorporated to protect sensitive personnel data. It also describes the system interfaces, highlighting how users interact with the system to perform various personnel management tasks such as employee registration, attendance tracking, leave management, payroll processing, and report generation.

In addition, this chapter outlines the hardware and software requirements necessary for running the system effectively and presents the testing strategies employed to ensure that the system meets specified functional and non-functional requirements. System testing results are discussed to demonstrate the reliability, accuracy, and efficiency of the implemented system.

Overall, this chapter provides evidence that the developed Personnel Management Information System successfully implements the proposed design and addresses the limitations identified in the existing manual personnel management system.

4.2 System Implementation

This section describes how the proposed Personnel Management Information System (PMIS) was developed and deployed based on the system design specified in Chapter Three. The implementation phase involved translating the architectural design, database schema, and process models into a functional software system. Emphasis was placed on modular development, data integrity, system security, and ease of use to ensure that the final system meets organizational requirements.

The system was implemented using a web-based approach, allowing authorized users to access personnel information through a browser interface. Each functional component—such as employee management, attendance tracking, leave processing, payroll computation, and report generation—was developed as an independent module to enhance maintainability and scalability. After development, the system was deployed in a controlled environment where it could be tested and evaluated for performance and reliability.

4.2.1 Development Environment

The development environment defines the technologies, tools, and platforms used in building the Personnel Management Information System. Appropriate technologies were selected based on reliability, ease of development, cost-effectiveness, and suitability for small- to medium-sized organizations.

Programming Language Used

The system was developed using Python as the primary programming language. Python was chosen due to its simplicity, readability, and strong support for rapid application development. It

also provides extensive libraries for database connectivity, web development, and security, making it suitable for developing a personnel management application.

Frameworks and Libraries

A web application framework was employed to structure the system and manage core functionalities such as routing, session management, and user authentication. Supporting libraries were used to handle form validation, database interaction, and report generation. The use of established frameworks and libraries improved development efficiency, enhanced code organization, and ensured adherence to best practices.

Database Management System

The system utilizes MySQL as the database management system. MySQL was selected because of its reliability, support for relational data modeling, and compatibility with web-based applications. It efficiently manages large volumes of personnel data and supports structured query operations required for report generation, payroll processing, and employee record management.

Development Tools and Integrated Development Environments (IDEs)

The system was developed using a modern Integrated Development Environment (IDE) that supports Python programming, code debugging, and version control. Additional development tools included:

- A web browser for interface testing
- Database management tools for database creation and administration
- Version control tools to track code changes and manage system updates

These tools collectively enhanced productivity, reduced development errors, and ensured smooth integration of system components.

4.2.2 Database Implementation

The database implementation phase involved translating the conceptual and logical database designs presented in Chapter Three into a physical database structure. This phase focused on creating the database, defining tables and relationships, inserting sample data, and establishing secure connectivity between the application and the database management system. The goal was to ensure efficient data storage, integrity, and easy retrieval of personnel information.

Database Creation Process

The database was created using the MySQL Database Management System. A dedicated database was designed specifically for the Personnel Management Information System to store all employee-related information. Structured Query Language (SQL) commands were used to create the database and define access privileges for authorized users. Proper naming conventions and constraints were applied to enhance database organization and security.

The database creation process ensured that the system could handle essential personnel data such as employee biodata, job roles, departments, attendance records, leave information, payroll details, and system user accounts.

Table Creation and Relationships

Several relational tables were created to represent the core entities of the personnel management system. Each table was designed with a primary key to uniquely identify records, while foreign keys were used to establish relationships between related tables. This approach minimized data redundancy and ensured data consistency.

Key tables implemented in the database include:

- **Employee Table**: Stores personal and employment details of staff
- **Department Table**: Contains department information within the organization
- **Attendance Table**: Records employee attendance data
- **Leave Table**: Manages employee leave requests and approvals
- **Payroll Table**: Stores salary and payment details
- **User Table**: Handles system authentication and authorization

Relationships were defined such that each employee is linked to a department, attendance and leave records are associated with specific employees, and payroll data references employee identifiers. Referential integrity constraints were enforced to prevent orphan records and maintain consistency across tables.

Sample Data Insertion

After creating the database tables, sample data was inserted to test system functionality and validate database operations. The sample records included employee profiles, departmental information, attendance logs, leave entries, and payroll records. This data enabled developers to simulate real-world usage scenarios, verify data retrieval accuracy, and ensure that system modules functioned as expected during testing.

Sample data insertion also supported the evaluation of report generation and payroll computation features of the system.

Database Connectivity

Database connectivity was established between the application and the MySQL database using a secure database connection interface supported by the chosen programming language and framework. Connection parameters such as database name, username, password, and host address were configured within the application environment.

The connectivity mechanism allows the system to perform essential database operations including data insertion, update, deletion, and retrieval. Security measures such as restricted access credentials and controlled database privileges were implemented to protect sensitive personnel data from unauthorized access.

4.2.3 Module Implementation

The Personnel Management Information System was implemented using a **modular approach**, where each major function of the system was developed as an independent but integrated module. This approach improves system maintainability, scalability, and ease of testing. Each module performs a specific task while interacting with the central database to ensure data consistency and integrity.

The major modules implemented in the system are described below.

User Authentication and Authorization Module

This module controls access to the system by ensuring that only authorized users can log in and perform operations based on their assigned roles. User credentials such as username and password are validated against stored records in the database.

Role-based access control was implemented to differentiate system privileges. For example, administrators have full access to manage personnel records and generate reports, while regular

staff users are restricted to viewing or updating limited information. This module enhances system security and protects sensitive personnel data.

Employee Registration and Management Module

The employee management module handles the creation, modification, retrieval, and deletion of employee records. It allows administrators to capture comprehensive employee information such as personal details, department, job designation, employment date, and contact information.

This module supports efficient searching and updating of employee records, eliminating the need for manual file handling. All employee-related data entered through this module is stored securely in the database for easy retrieval and reporting.

Attendance Management Module

The attendance module records and manages daily staff attendance. It allows authorized users to mark attendance, view attendance history, and generate attendance summaries for specific periods.

Attendance data captured through this module supports administrative monitoring and serves as an input for payroll processing. Automating attendance management reduces errors associated with manual attendance registers and improves record accuracy.

Leave Management Module

The leave management module enables employees to submit leave requests electronically and allows administrators to review, approve, or reject such requests. The system tracks leave types, leave duration, approval status, and remaining leave balance.

This module ensures transparency in leave administration and provides accurate leave records that can be referenced for workforce planning and payroll adjustments.

Payroll Processing Module

The payroll module is responsible for calculating employee salaries based on predefined parameters such as basic salary, allowances, deductions, attendance, and approved leave records. The module automatically generates payroll records, reducing calculation errors associated with manual payroll processing.

Payroll information generated by this module can be viewed, stored, and included in summary reports, ensuring timely and accurate salary administration.

Report Generation Module

The report generation module produces various system reports required by management. These include employee lists, attendance summaries, leave reports, and payroll reports.

Reports can be generated based on selected criteria such as department, date range, or employee category. This module supports informed decision-making by providing timely and accurate personnel information in an organized format.

4.3 System Interface

This section presents the user interfaces of the developed Personnel Management Information System. The system interfaces were designed to be simple, intuitive, and user-friendly, ensuring that users can perform their tasks efficiently with minimal training. Emphasis was placed on clarity, consistency, and ease of navigation across all system modules.

Screenshots of the implemented interfaces are included in this section to illustrate how users interact with the system and to demonstrate the practical realization of the design specifications discussed in Chapter Three.

Login Interface

The login interface serves as the entry point to the system. It requires users to enter valid authentication credentials, such as a username and password, before gaining access. This interface ensures system security by preventing unauthorized access.

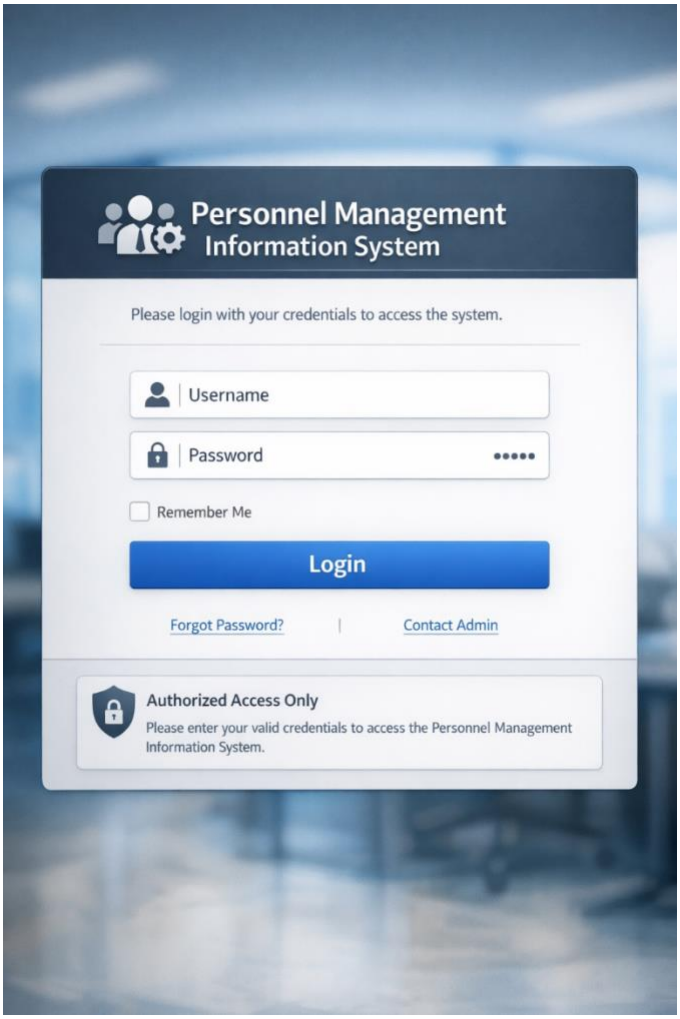

Fig 4.1: login screenshot

Dashboard Interface

The dashboard interface provides users with an overview of the system after successful login. It displays key information and quick access links to major system modules such as employee management, attendance, leave, payroll, and reports.

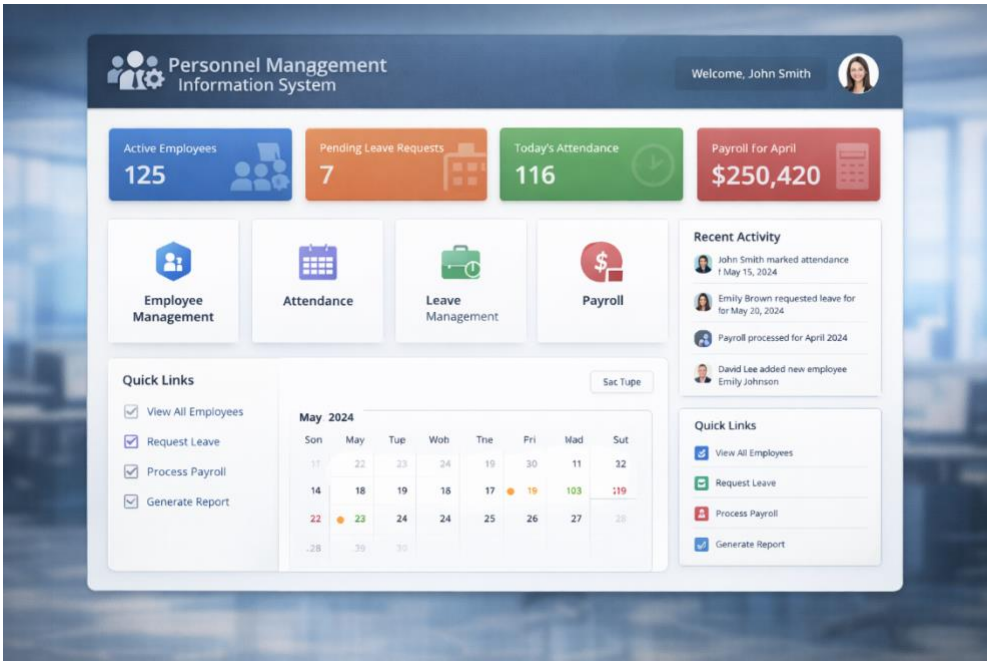

Fig 4.2: screenshot of PMIS Dashboard.

Employee Registration and Management Interface

This interface allows administrators to register new employees and manage existing employee records. It includes structured input fields for capturing personal details, department, job role, and employment information.

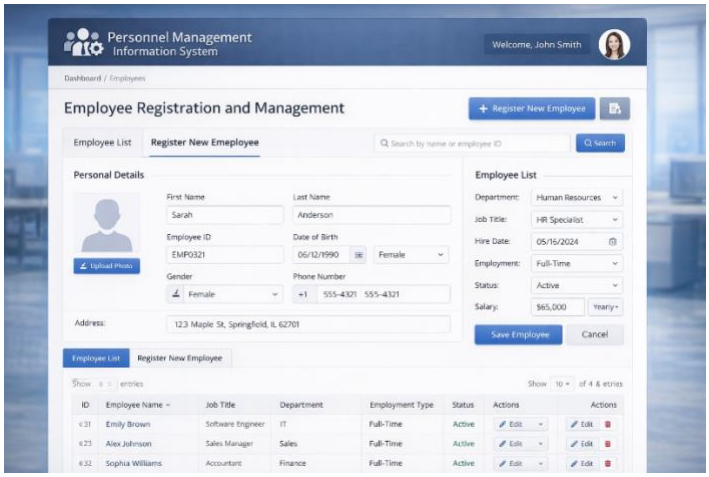

Fig 4.3: screenshot of Employee Registration and Management Interface

Attendance Management Interface

The attendance interface enables authorized users to record and view employee attendance records.

Users can select dates, mark attendance, and retrieve attendance history.

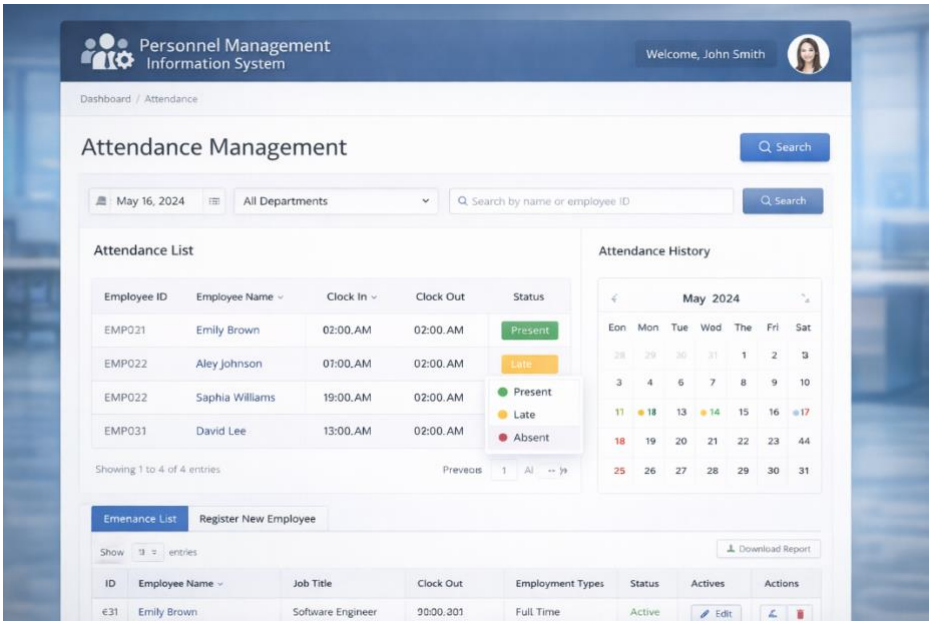

Fig 4.4: screenshot of Attendance Management Interface

Leave Management Interface

This interface supports the submission, review, and approval of employee leave requests. Administrators can monitor leave status and manage leave balances.

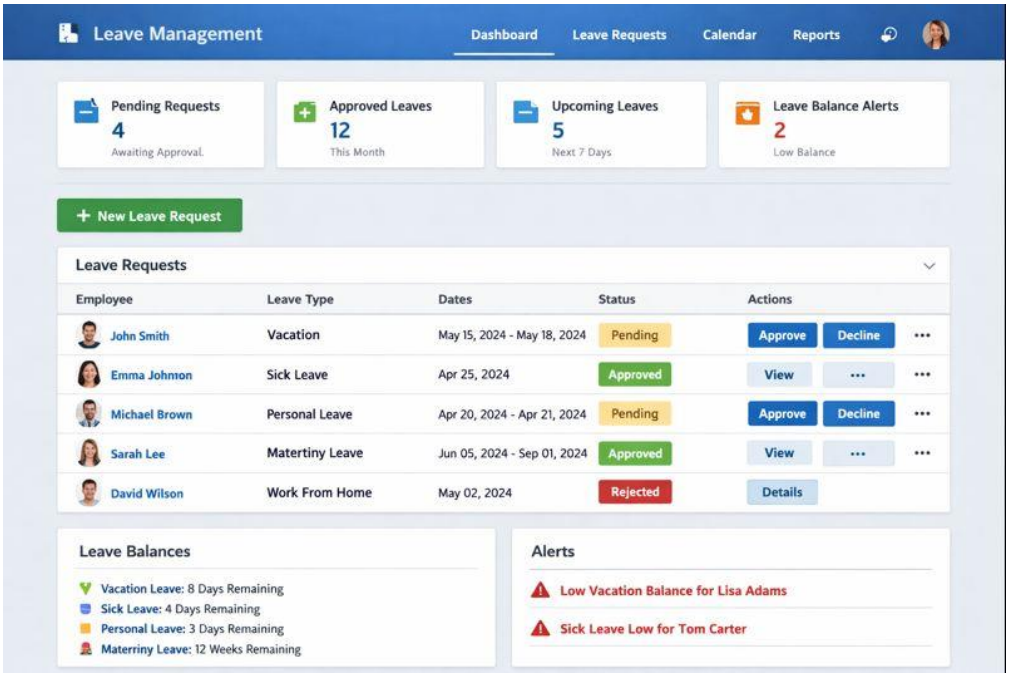

Fig 4.5: screenshot of Leave Management Interface

Payroll Interface

The payroll interface facilitates salary computation and payroll management. It allows administrators to generate payroll records based on attendance and approved leave data.

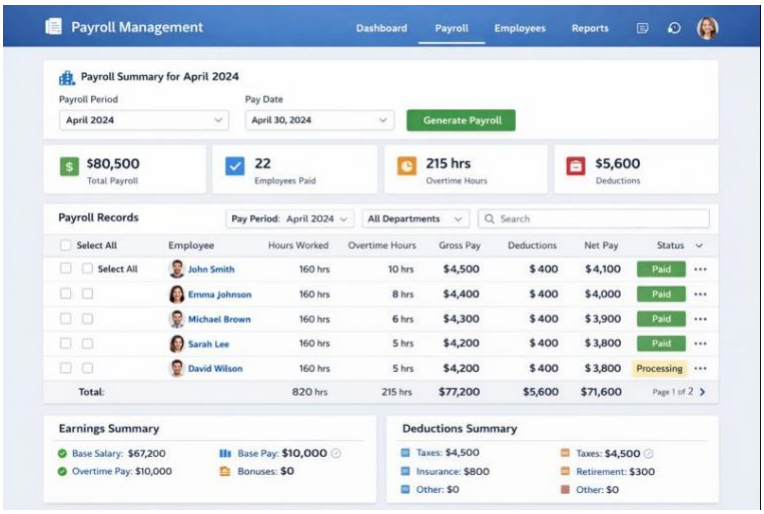

Fig 4.6: screenshoot of Payroll Interface

Report Generation Interface

The report generation interface enables users to produce various personnel-related reports. Users can specify criteria such as date range or department before generating reports.

Administrative Interface

The administrative interface allows system administrators to manage users, assign roles, and control system settings.

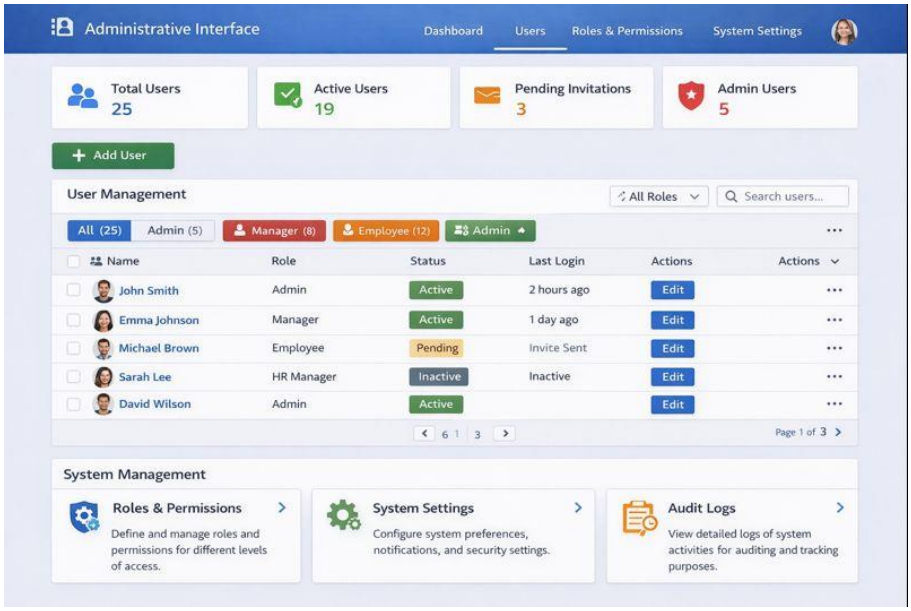

Fig 4.7: screenshot of Administrative Interface

4.4 System Testing

System testing was carried out to ensure that the developed Personnel Management Information System functions correctly and meets the specified requirements. The testing process focused on verifying system reliability, accuracy, security, and usability. Various testing techniques were applied to identify errors, validate system performance, and confirm that the system fulfills both functional and non-functional requirements.

Testing was conducted after the completion of system implementation and before final deployment to ensure that all modules operate as expected under normal and abnormal conditions.

4.4.1 Testing Strategy

A structured testing strategy was adopted to evaluate the system comprehensively. The following testing methods were employed:

- **Unit Testing**: Individual system modules such as login, employee registration, attendance, leave, payroll, and report generation were tested separately to ensure that each module performs its intended function correctly.
- **Integration Testing**: This testing verified that different modules interact correctly with one another, particularly the interaction between the attendance, leave, and payroll modules.
- **System Testing**: The complete system was tested as a whole to ensure that all components function together seamlessly in a real-world operational environment.
- **User Acceptance Testing (UAT)**: Selected users tested the system to confirm that it meets user expectations, is easy to use, and satisfies organizational requirements.

4.4.2 Test Cases

Several test cases were designed to validate system functionality. Each test case defined specific input conditions, expected outputs, and actual outcomes. Key test scenarios included:

- Valid and invalid user login attempts
- Employee record creation and modification
- Attendance marking and retrieval
- Leave request submission and approval
- Payroll computation accuracy
- Report generation based on selected criteria

These test cases ensured that the system responds correctly to both normal and erroneous inputs.

4.4.3 Test Results

The results of the system testing showed that the Personnel Management Information System performed effectively across all tested scenarios. Identified errors were corrected during the testing phase, and subsequent retesting confirmed that the issues had been resolved.

The system demonstrated accurate data processing, reliable performance, and improved efficiency compared to the existing manual system. User feedback from acceptance testing indicated that the system is easy to use and meets the operational needs of personnel management within the organization.

CHAPTER FIVE

SUMMARY, CONCLUSION AND RECOMMENDATIONS

5.1 SUMMARY OF FINDINGS

The primary objective of this project was to design and implement a computerized Personnel Management Information System capable of addressing the limitations of the existing manual personnel management process. From the analysis carried out in earlier chapters, it was discovered that the manual system was characterized by inefficiency, data redundancy, slow information retrieval, poor record security, and difficulty in generating timely reports.

The developed system successfully automated personnel operations such as employee record management, attendance tracking, leave administration, payroll processing, and report generation. The implementation of a centralized database improved data accuracy, reduced redundancy, and enhanced accessibility of personnel information. Testing results further confirmed that the system met both functional and non-functional requirements, demonstrating improved efficiency, reliability, and ease of use compared to the manual system.

5.2 Conclusion

Based on the objectives set at the beginning of the study and the results obtained from system implementation and testing, it can be concluded that the developed Personnel Management Information System effectively addresses the challenges associated with manual personnel management. The system enhances efficiency, accuracy, data security, and decision-making within the organization.

The project successfully achieved its aim of designing and implementing a functional, user-friendly, and reliable personnel management system, thereby validating the relevance of computerized solutions in modern organizational management.

5.3 Recommendations

Based on the findings of this study, the following recommendations are made:

- Organizations should adopt computerized personnel management systems to improve efficiency and data accuracy
- Proper user training should be conducted before full system deployment
- Regular data backup and system maintenance should be enforced to ensure data security
- Management should enforce access control policies to protect sensitive personnel information

5.4 Suggestions for Further Work

Although the developed system meets its intended objectives, future enhancements can be considered, including:

- Development of a mobile-based version of the system
- Integration with biometric attendance systems
- Deployment of the system on a cloud platform
- Incorporation of advanced analytics and reporting tools
- Integration with other organizational systems such as accounting and project management systems

REFERENCES

Adebayo, I. O. (2019). Design and implementation of a web-based staff information management system. *International Journal of Computer Science and Information Technology Research, 7*(1), 71-84.

Alam, M. M. D., Alam, M. Z., Rahman, S. A., & Taghizadeh, S. K. (2020). Factors influencing mHealth adoption and its impact on mental well-being during COVID-19 pandemic. *Journal of Biomedical Informatics, 116*, 103722. https://doi.org/10.1016/j.jbi.2021.103722

Apache Software Foundation. (2021). *Apache NetBeans IDE*. Retrieved from https://netbeans.apache.org/

Armstrong, M., & Taylor, S. (2020). *Armstrong's handbook of human resource management practice* (15th ed.). Kogan Page Publishers.

Armstrong, M., & Taylor, S. (2020). *Armstrong's handbook of human resource management practice* (15th ed.). Kogan Page Publishers.

Balaji, S., & Murugaiyan, M. S. (2020). Waterfall vs. V-Model vs. Agile: A comparative study on SDLC. *International Journal of Information Technology and Business Management, 2*(1), 26-30.

BambooHR. (2021). *BambooHR: HR software for small and medium businesses*. Retrieved from https://www.bamboohr.com/

Banks, A., & Porcello, E. (2020). *Learning React: Modern patterns for developing React apps* (2nd ed.). O'Reilly Media.

Barney, J. B., & Hesterly, W. S. (2019). *Strategic management and competitive advantage: Concepts and cases* (6th ed.). Pearson Education.

Beck, K. (2020). *Extreme programming explained: Embrace change* (2nd ed.). Addison-Wesley Professional.

Becker, G. S. (2020). *Human capital: A theoretical and empirical analysis with special reference to education* (3rd ed.). University of Chicago Press.

Bharadwaj, A., El Sawy, O. A., Pavlou, P. A., & Venkatraman, N. (2020). Digital business strategy: Toward a next generation of insights. *MIS Quarterly, 37*(2), 471-482.

Boehm, B. W. (2020). A spiral model of software development and enhancement. *IEEE Computer, 21*(5), 61-72.

Bondarouk, T., Parry, E., & Furtmueller, E. (2019). Electronic HRM: Four decades of research on adoption and consequences. *The International Journal of Human Resource Management, 30*(2-3), 214-239. https://doi.org/10.1080/09585192.2018.1521534

Boselie, P. (2019). *Strategic human resource management: A balanced approach* (3rd ed.). McGraw-Hill Education.

Boudreau, J. W., & Cascio, W. F. (2020). Human capital analytics: Why are we not there? *Journal of Organizational Effectiveness: People and Performance, 7*(1), 51-61. https://doi.org/10.1108/JOEPP-03-2019-0029

Campanelli, A. S., & Parreiras, F. S. (2020). Agile methods tailoring: A systematic literature review. *Journal of Systems and Software, 110*, 85-100.

Carnevale, J. B., & Hatak, I. (2020). Employee adjustment and well-being in the era of COVID-19: Implications for human resource management. *Journal of Business Research, 116*, 183-187. https://doi.org/10.1016/j.jbusres.2020.05.037

Cascio, W. F., & Boudreau, J. W. (2019). *Investing in people: Financial impact of human resource initiatives* (3rd ed.). Society for Human Resource Management.

Colombo, G., Dell'Era, C., & Frattini, F. (2019). New product development (NPD) service suppliers in open innovation practices. *International Journal of Innovation Management, 19*(3), 1550034.

Coronel, C., & Morris, S. (2019). *Database systems: Design, implementation, & management* (13th ed.). Cengage Learning.

Date, C. J. (2019). *Database design and relational theory: Normal forms and all that jazz* (2nd ed.). Apress.

Dessler, G. (2020). *Human resource management* (16th ed.). Pearson Education.

Dubois, P. (2020). *MySQL* (5th ed.). Addison-Wesley Professional.

Elmasri, R., & Navathe, S. B. (2020). *Fundamentals of database systems* (7th ed.). Pearson Education.

Forcier, J., Bissex, P., & Chun, W. J. (2020). *Python web development with Django*. Addison-Wesley Professional.

Furnell, S. (2020). The challenge of user authentication and the evolution of user-facing security technologies. *Computers & Security, 100*, 102059. https://doi.org/10.1016/j.cose.2020.102059

Grassi, P. A., Garcia, M. E., & Fenton, J. L. (2020). *Digital identity guidelines: Authentication and lifecycle management* (NIST Special Publication 800-63B). National Institute of Standards and Technology.

Gupta, S., & Seetharaman, P. (2020). The usage and adoption of cloud computing by small and medium businesses. *International Journal of Information Management, 33*(5), 861-874.

Hahn, E. (2020). *Express in action: Node applications with Express and its ecosystem*. Manning Publications.

Haverbeke, M. (2020). *Eloquent JavaScript: A modern introduction to programming* (3rd ed.). No Starch Press.

Hoofnagle, C. J., van der Sloot, B., & Borgesius, F. Z. (2019). The European Union general data protection regulation: What it is and what it means. *Information & Communications Technology Law, 28*(1), 65-98. https://doi.org/10.1080/13600834.2019.1573501

IceHRM. (2021). *IceHRM: Human resource management system*. Retrieved from https://icehrm.com/

IJetBrains. (2021). *PyCharm: The Python IDE for professional developers*. Retrieved from https://www.jetbrains.com/pycharm/

Iqbal, N., Ahmad, M., Allen, M. M., & Raziq, M. M. (2019). Does e-HRM improve labour productivity? A study of commercial bank workplaces in Pakistan. *Employee Relations: The International Journal, 41*(3), 542-562. https://doi.org/10.1108/ER-01-2018-0035

Johnson, R. D., Stone, D. L., & Lukaszewski, K. M. (2020). The benefits of eHRM and AI for talent acquisition. *Journal of Tourism Futures, 7*(1), 40-52.

Kavanagh, M. J., & Johnson, R. D. (Eds.). (2021). *Human resource information systems: Basics, applications, and future directions* (5th ed.). SAGE Publications.

Khan, S., Khan, S., & Aftab, M. (2020). Digitization and its impact on economy. *International Journal of Digital Library Services, 5*(1), 138-149.

Laudon, K. C., & Laudon, J. P. (2020). *Management information systems: Managing the digital firm* (16th ed.). Pearson Education.

Lazar, J., Feng, J. H., & Hochheiser, H. (2020). *Research methods in human-computer interaction* (2nd ed.). Morgan Kaufmann.

Malik, A., Budhwar, P., Mohan, H., & Kumar, S. (2021). Augmented human resource management: A systematic review and research agenda. *Human Resource Management Review, 31*(4), 100848. https://doi.org/10.1016/j.hrmr.2021.100848

Marangunić, N., & Granić, A. (2019). Technology acceptance model: A literature review from 1986 to 2013. *Universal Access in the Information Society, 14*(1), 81-95.

Mardiana, S., Tjakraatmadja, J. H., & Aprianingsih, A. (2020). Validating the conceptual model for predicting intention to use as part of information system success model: The case of an Indonesian government agency. *Procedia Computer Science, 72*, 353-360.

Marler, J. H., & Boudreau, J. W. (2021). An evidence-based review of HR Analytics. *The International Journal of Human Resource Management, 28*(1), 3-26. https://doi.org/10.1080/09585192.2016.1244699

McKinney, W. (2020). *Python for data analysis: Data wrangling with Pandas, NumPy, and IPython* (2nd ed.). O'Reilly Media.

Mele, C., Russo-Spena, T., Kaartemo, V., & Marzullo, M. L. (2020). Smart nudging: How cognitive technologies enable choice architectures for value co-creation. *Journal of Business Research, 129*, 949-960.

Mention, A. L., & Bontis, N. (2019). Intellectual capital and performance within the banking sector of Luxembourg and Belgium. *Journal of Intellectual Capital, 14*(2), 286-309.

Meyer, E. A. (2021). *CSS: The definitive guide: Visual presentation for the web* (5th ed.). O'Reilly Media.

Microsoft. (2021). *SQL Server 2019*. Retrieved from https://www.microsoft.com/en-us/sql-server/

Mondy, R. W., & Martocchio, J. J. (2020). *Human resource management* (15th ed.). Pearson Education.

Munirat, Y., Soriyan, H. A., & Salaudeen, B. (2020). Design and implementation of automated human resource management system. *International Journal of Computer Applications, 975*, 8887.

Noe, R. A., Hollenbeck, J. R., Gerhart, B., & Wright, P. M. (2021). *Human resource management: Gaining a competitive advantage* (12th ed.). McGraw-Hill Education.

Nura, A. A., & Osman, N. H. (2019). The influence of human resource information system (HRIS) application on human resource management (HRM) strategic role: Evidence from the Nigerian public sector. *Journal of Engineering and Applied Sciences, 14*(3), 798-808. https://doi.org/10.36478/jeasci.2019.798.808

Obeidat, S. M. (2020). The impact of human resource information systems (HRIS) on strategic human resource management: Evidence from commercial banks in Jordan. *International Journal of Business and Social Science, 11*(9), 13-22. https://doi.org/10.30845/ijbss.v11n9p2

Odoo. (2021). *Odoo: Open source ERP and CRM*. Retrieved from https://www.odoo.com/

Oladele, T. O., & Adeoti, O. A. (2021). Personnel information management system with advanced security features. *Journal of Computer Science and Applications, 8*(2), 45-56.

Oracle Corporation. (2021). *Oracle HCM Cloud*. Retrieved from https://www.oracle.com/human-capital-management/

OrangeHRM. (2021). *OrangeHRM: Open source HR management system*. Retrieved from https://www.orangehrm.com/

Otwell, T. (2021). *Laravel: The PHP framework for web artisans*. Retrieved from https://laravel.com/

OWASP Foundation. (2021). *OWASP Top Ten Project*. Retrieved from https://owasp.org/www-project-top-ten/

Oz, E., & Jones, A. (2020). *Management information systems* (8th ed.). Cengage Learning.

Preece, J., Sharp, H., & Rogers, Y. (2019). *Interaction design: Beyond human-computer interaction* (5th ed.). John Wiley & Sons.

Pressman, R. S., & Maxim, B. R. (2020). *Software engineering: A practitioner's approach* (9th ed.). McGraw-Hill Education.

Rescorla, E. (2020). *The transport layer security (TLS) protocol version 1.3* (RFC 8446). Internet Engineering Task Force.

Robbins, J. N. (2020). *Learning web design: A beginner's guide to HTML, CSS, JavaScript, and web graphics* (5th ed.). O'Reilly Media.

Rousseau, D. M. (2019). Organizational behavior in the new organizational era. *Annual Review of Psychology, 48*(1), 515-546.

Sandhu, R., Ferraiolo, D., & Kuhn, R. (2020). The NIST model for role-based access control. *ACM Transactions on Information and System Security, 3*(3), 224-274.

SAP. (2021). *SAP SuccessFactors*. Retrieved from https://www.sap.com/products/human-resources-hcm.html

Schwaber, K., & Sutherland, J. (2020). *The Scrum guide: The definitive guide to Scrum*. Scrum.org.

Sharma, A., & Sharma, T. (2020). HR analytics and performance appraisal system: A conceptual framework for employee performance improvement. *Management Research Review, 43*(10), 1299-1322. https://doi.org/10.1108/MRR-04-2019-0154

Sierra, K., & Bates, B. (2020). *Head first Java* (3rd ed.). O'Reilly Media.

Skyttner, L. (2020). *General systems theory: Problems, perspectives, practice* (3rd ed.). World Scientific Publishing.

Smith, G. (2020). *PostgreSQL: Up and running* (3rd ed.). O'Reilly Media.

Sommerville, I. (2020). *Software engineering* (10th ed.). Pearson Education.

Spinellis, D. (2020). *Effective debugging: 66 specific ways to debug software and systems*. Addison-Wesley Professional.

SQLite. (2021). *SQLite: Small, fast, self-contained SQL database engine*. Retrieved from https://www.sqlite.org/

Stair, R., & Reynolds, G. (2020). *Principles of information systems* (14th ed.). Cengage Learning.

Stone, D. L., Deadrick, D. L., Lukaszewski, K. M., & Johnson, R. (2020). The influence of technology on the future of human resource management. *Human Resource Management Review, 25*(2), 216-231. https://doi.org/10.1016/j.hrmr.2015.01.002

Strohmeier, S., & Parry, E. (2020). HRM in the digital age–digital changes and challenges of the HR profession. *Employee Relations: The International Journal, 42*(4), 345-361. https://doi.org/10.1108/ER-03-2019-0101

Stuttard, D., & Pinto, M. (2019). *The web application hacker's handbook: Finding and exploiting security flaws* (2nd ed.). John Wiley & Sons.

Tatroe, K., MacIntyre, P., & Lerdorf, R. (2020). *Programming PHP: Creating dynamic web pages* (4th ed.). O'Reilly Media.

Teece, D. J. (2019). A dynamic capabilities-based entrepreneurial theory of the multinational enterprise. *Journal of International Business Studies, 45*(1), 8-37.

Tushman, M. L., & Nadler, D. A. (2019). Information processing as an integrating concept in organizational design. *Academy of Management Review, 3*(3), 613-624.

Venkatesh, V., Morris, M. G., Davis, G. B., & Davis, F. D. (2020). User acceptance of information technology: Toward a unified view. *MIS Quarterly, 27*(3), 425-478.

Walls, C. (2020). *Spring Boot in action*. Manning Publications.

Weissbacher, M., Robertson, W., Kirda, E., Kruegel, C., & Vigna, G. (2020). ZigZag: Automatically hardening web applications against client-side validation vulnerabilities. *Proceedings of the 29th USENIX Security Symposium*, 83-98.

Whitman, M. E., & Mattord, H. J. (2021). *Principles of information security* (7th ed.). Cengage Learning.

Workday. (2021). *Workday HCM*. Retrieved from https://www.workday.com/en-us/products/human-capital-management/overview.html

Wright, P. M., & McMahan, G. C. (2020). Theoretical perspectives for strategic human resource management. *Journal of Management, 18*(2), 295-320.

Wright, P. M., Coff, R., & Moliterno, T. P. (2021). Strategic human capital: Crossing the great divide. *Journal of Management, 40*(2), 353-370.

Zoho Corporation. (2021). *Zoho People: Cloud-based HR management software*. Retrieved from https://www.zoho.com/people/

APPENDIX SYSTEM SOURCE CODE

This appendix presents the complete source code of the Personnel Management Information System (PMIS) developed using Python (Flask framework) and MySQL. The code is organized by module for clarity and ease of understanding.

E1. Database Configuration (config.py)

This module defines the database connection settings and establishes connectivity between the Flask application and the MySQL database.

```
# config.py

# Database and application configuration settings


import os


class Config:

    SECRET_KEY = os.environ.get('SECRET_KEY') or 'pmis_secret_key_2025'

    MYSQL_HOST = 'localhost'

    MYSQL_USER = 'root'

    MYSQL_PASSWORD = 'pmis_password'

    MYSQL_DB = 'pmis_db'

    MYSQL_CURSORCLASS = 'DictCursor'

    DEBUG = False


class DevelopmentConfig(Config):

    DEBUG = True


class ProductionConfig(Config):

    DEBUG = False
```

```
config = {
    'development': DevelopmentConfig,
    'production': ProductionConfig,
    'default': DevelopmentConfig
}
```

E2. Database Schema (schema.sql)

This SQL script creates the PMIS database and all required relational tables with appropriate constraints and relationships.

```
-- schema.sql
-- PMIS Database Schema

CREATE DATABASE IF NOT EXISTS pmis_db;
USE pmis_db;

-- Department Table
CREATE TABLE department (
    dept_id INT AUTO_INCREMENT PRIMARY KEY,
    dept_name VARCHAR(100) NOT NULL,
    dept_head VARCHAR(100),
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

-- Employee Table
CREATE TABLE employee (
    emp_id INT AUTO_INCREMENT PRIMARY KEY,
    first_name VARCHAR(50) NOT NULL,
    last_name VARCHAR(50) NOT NULL,
    date_of_birth DATE,
    gender ENUM('Male', 'Female', 'Other'),
    phone VARCHAR(20),
    email VARCHAR(100) UNIQUE,
```

```
    address TEXT,
    dept_id INT,
    job_title VARCHAR(100),
    employment_date DATE,
    salary DECIMAL(10, 2) DEFAULT 0.00,
    status ENUM('Active', 'Inactive') DEFAULT 'Active',
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (dept_id) REFERENCES department(dept_id)
        ON DELETE SET NULL ON UPDATE CASCADE
);

-- User Authentication Table
CREATE TABLE users (
    user_id INT AUTO_INCREMENT PRIMARY KEY,
    username VARCHAR(50) NOT NULL UNIQUE,
    password_hash VARCHAR(255) NOT NULL,
    emp_id INT,
    role ENUM('Admin', 'HR', 'Staff') DEFAULT 'Staff',
    is_active BOOLEAN DEFAULT TRUE,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (emp_id) REFERENCES employee(emp_id)
        ON DELETE SET NULL ON UPDATE CASCADE
);

-- Attendance Table
```

```
CREATE TABLE attendance (
   att_id INT AUTO_INCREMENT PRIMARY KEY,
   emp_id INT NOT NULL,
   att_date DATE NOT NULL,
   check_in TIME,
   check_out TIME,
   status ENUM('Present', 'Absent', 'Late', 'Half-Day') DEFAULT 'Present',
   remarks VARCHAR(200),
   FOREIGN KEY (emp_id) REFERENCES employee(emp_id)
      ON DELETE CASCADE ON UPDATE CASCADE,
   UNIQUE KEY unique_att (emp_id, att_date)
);

-- Leave Table
CREATE TABLE leave_request (
   leave_id INT AUTO_INCREMENT PRIMARY KEY,
   emp_id INT NOT NULL,
   leave_type ENUM('Annual', 'Sick', 'Maternity', 'Paternity',
                   'Emergency', 'Unpaid') NOT NULL,
   start_date DATE NOT NULL,
   end_date DATE NOT NULL,
   num_days INT NOT NULL,
   reason TEXT,
   status ENUM('Pending', 'Approved', 'Rejected') DEFAULT 'Pending',
   applied_on TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
```

```
    reviewed_by INT,
    reviewed_on TIMESTAMP,
    FOREIGN KEY (emp_id) REFERENCES employee(emp_id)
        ON DELETE CASCADE ON UPDATE CASCADE
);


-- Payroll Table
CREATE TABLE payroll (
    payroll_id INT AUTO_INCREMENT PRIMARY KEY,
    emp_id INT NOT NULL,
    pay_period VARCHAR(20) NOT NULL,
    basic_salary DECIMAL(10, 2) NOT NULL,
    housing_allowance DECIMAL(10, 2) DEFAULT 0.00,
    transport_allowance DECIMAL(10, 2) DEFAULT 0.00,
    other_allowances DECIMAL(10, 2) DEFAULT 0.00,
    tax_deduction DECIMAL(10, 2) DEFAULT 0.00,
    other_deductions DECIMAL(10, 2) DEFAULT 0.00,
    net_salary DECIMAL(10, 2) NOT NULL,
    payment_status ENUM('Pending', 'Paid') DEFAULT 'Pending',
    generated_on TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (emp_id) REFERENCES employee(emp_id)
        ON DELETE CASCADE ON UPDATE CASCADE
);
```

E3. Application Entry Point (app.py)

This is the main Flask application file that initializes the app, registers blueprints, and configures the database connection using Flask-MySQLdb.

```
# app.py

# Main application entry point


from flask import Flask

from flask_mysqldb import MySQL

from config import config


mysql = MySQL()


def create_app(config_name='default'):

    app = Flask(__name__)

    app.config.from_object(config[config_name])


    mysql.init_app(app)


    # Register Blueprints

    from auth.routes import auth_bp

    from employees.routes import emp_bp

    from attendance.routes import att_bp

    from leave.routes import leave_bp

    from payroll.routes import payroll_bp

    from reports.routes import report_bp
```

```
    app.register_blueprint(auth_bp)
    app.register_blueprint(emp_bp, url_prefix='/employees')
    app.register_blueprint(att_bp, url_prefix='/attendance')
    app.register_blueprint(leave_bp, url_prefix='/leave')
    app.register_blueprint(payroll_bp, url_prefix='/payroll')
    app.register_blueprint(report_bp, url_prefix='/reports')

    return app

if __name__ == '__main__':
    app = create_app('development')
    app.run(debug=True)
```

E4. Authentication Module (auth/routes.py)

This module handles user login, logout, and session management. It validates credentials against the database and enforces role-based access control.

```
# auth/routes.py
# User authentication and session management

from flask import (Blueprint, render_template, request,
                   redirect, url_for, session, flash)
from werkzeug.security import check_password_hash, generate_password_hash
from app import mysql
import functools

auth_bp = Blueprint('auth', __name__)

# Login required decorator
def login_required(f):
    @functools.wraps(f)
    def decorated_function(*args, **kwargs):
        if 'user_id' not in session:
            flash('Please log in to access this page.', 'warning')
            return redirect(url_for('auth.login'))
        return f(*args, **kwargs)
    return decorated_function

# Admin only decorator
def admin_required(f):
```

```
    @functools.wraps(f)
    def decorated_function(*args, **kwargs):
        if session.get('role') not in ['Admin', 'HR']:
            flash('Access denied. Insufficient privileges.', 'danger')
            return redirect(url_for('dashboard'))
        return f(*args, **kwargs)
    return decorated_function


@auth_bp.route('/', methods=['GET', 'POST'])
@auth_bp.route('/login', methods=['GET', 'POST'])
def login():
    if 'user_id' in session:
        return redirect(url_for('dashboard'))
    if request.method == 'POST':
        username = request.form.get('username', '').strip()
        password = request.form.get('password', '')
        cur = mysql.connection.cursor()
        cur.execute(
            'SELECT * FROM users WHERE username = %s AND is_active = 1',
            (username,)
        )
        user = cur.fetchone()
        cur.close()
        if user and check_password_hash(user['password_hash'], password):
            session['user_id'] = user['user_id']
```

```
            session['username'] = user['username']
            session['role'] = user['role']
            session['emp_id'] = user['emp_id']
            flash(f"Welcome, {user['username']}!", 'success')
            return redirect(url_for('dashboard'))
        else:
            flash('Invalid username or password.', 'danger')
    return render_template('auth/login.html')


@auth_bp.route('/logout')
def logout():
    session.clear()
    flash('You have been logged out successfully.', 'info')
    return redirect(url_for('auth.login'))


@auth_bp.route('/dashboard')
@login_required
def dashboard():
    cur = mysql.connection.cursor()
    cur.execute('SELECT COUNT(*) AS total FROM employee WHERE status=%s',
('Active',))
    total_emp = cur.fetchone()['total']
    cur.execute('SELECT COUNT(*) AS total FROM leave_request WHERE status=%s',
('Pending',))
    pending_leave = cur.fetchone()['total']
```

```
    cur.execute('SELECT COUNT(*) AS total FROM attendance WHERE
att_date=CURDATE()')
    today_att = cur.fetchone()['total']
    cur.close()
    return render_template('dashboard.html',
                           total_emp=total_emp,
                           pending_leave=pending_leave,
                           today_att=today_att)
```

E5. Employee Management Module (employees/routes.py)

This module handles CRUD operations for employee records — registration, viewing, updating, and deactivating employee profiles.

```
# employees/routes.py

# Employee registration and management


from flask import (Blueprint, render_template, request,

                   redirect, url_for, flash)

from app import mysql

from auth.routes import login_required, admin_required


emp_bp = Blueprint('employees', __name__)


@emp_bp.route('/')

@login_required

def list_employees():

    cur = mysql.connection.cursor()

    cur.execute('''

        SELECT e.*, d.dept_name

        FROM employee e

        LEFT JOIN department d ON e.dept_id = d.dept_id

        ORDER BY e.last_name, e.first_name

    ''')

    employees = cur.fetchall()

    cur.close()

    return render_template('employees/list.html', employees=employees)
```

```
@emp_bp.route('/register', methods=['GET', 'POST'])
@login_required
@admin_required
def register_employee():
    cur = mysql.connection.cursor()
    cur.execute('SELECT * FROM department ORDER BY dept_name')
    departments = cur.fetchall()
    if request.method == 'POST':
        first_name = request.form['first_name'].strip()
        last_name = request.form['last_name'].strip()
        dob = request.form['date_of_birth']
        gender = request.form['gender']
        phone = request.form['phone']
        email = request.form['email']
        address = request.form['address']
        dept_id = request.form['dept_id']
        job_title = request.form['job_title']
        emp_date = request.form['employment_date']
        salary = request.form['salary']
        cur.execute('''
            INSERT INTO employee (first_name, last_name, date_of_birth,
            gender, phone, email, address, dept_id, job_title,
            employment_date, salary)
            VALUES (%s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s)
```

```
        ''', (first_name, last_name, dob, gender, phone, email,
              address, dept_id, job_title, emp_date, salary))
        mysql.connection.commit()
        cur.close()
        flash('Employee registered successfully.', 'success')
        return redirect(url_for('employees.list_employees'))
    cur.close()
    return render_template('employees/register.html', departments=departments)


@emp_bp.route('/edit/<int:emp_id>', methods=['GET', 'POST'])
@login_required
@admin_required
def edit_employee(emp_id):
    cur = mysql.connection.cursor()
    cur.execute('SELECT * FROM employee WHERE emp_id = %s', (emp_id,))
    employee = cur.fetchone()
    cur.execute('SELECT * FROM department ORDER BY dept_name')
    departments = cur.fetchall()
    if request.method == 'POST':
        cur.execute('''
            UPDATE employee SET first_name=%s, last_name=%s,
            phone=%s, email=%s, dept_id=%s, job_title=%s,
            salary=%s, status=%s WHERE emp_id=%s
        ''', (request.form['first_name'], request.form['last_name'],
              request.form['phone'], request.form['email'],
```

```
            request.form['dept_id'], request.form['job_title'],
            request.form['salary'], request.form['status'], emp_id))
        mysql.connection.commit()
        cur.close()
        flash('Employee record updated successfully.', 'success')
        return redirect(url_for('employees.list_employees'))
    cur.close()
    return render_template('employees/edit.html',
                           employee=employee, departments=departments)


@emp_bp.route('/view/<int:emp_id>')
@login_required
def view_employee(emp_id):
    cur = mysql.connection.cursor()
    cur.execute('''
        SELECT e.*, d.dept_name FROM employee e
        LEFT JOIN department d ON e.dept_id = d.dept_id
        WHERE e.emp_id = %s
    ''', (emp_id,))
    employee = cur.fetchone()
    cur.close()
    return render_template('employees/view.html', employee=employee)
```

E6. Attendance Management Module (attendance/routes.py)

This module manages daily attendance recording, retrieval, and summary reporting for all employees.

```
# attendance/routes.py
# Attendance tracking and management

from flask import (Blueprint, render_template, request,
                   redirect, url_for, flash)
from app import mysql
from auth.routes import login_required, admin_required
from datetime import date

att_bp = Blueprint('attendance', __name__)

@att_bp.route('/', methods=['GET', 'POST'])
@login_required
@admin_required
def mark_attendance():
    today = date.today().strftime('%Y-%m-%d')
    cur = mysql.connection.cursor()
    cur.execute('''
        SELECT e.emp_id, e.first_name, e.last_name, d.dept_name,
        a.status AS att_status
        FROM employee e
        LEFT JOIN department d ON e.dept_id = d.dept_id
        LEFT JOIN attendance a ON e.emp_id = a.emp_id
```

```
        AND a.att_date = CURDATE()
    WHERE e.status = 'Active'
    ORDER BY e.last_name
''')
employees = cur.fetchall()
if request.method == 'POST':
    for emp in employees:
        emp_id = emp['emp_id']
        status = request.form.get(f'status_{emp_id}', 'Absent')
        check_in = request.form.get(f'checkin_{emp_id}', None)
        check_out = request.form.get(f'checkout_{emp_id}', None)
        cur.execute('''
            INSERT INTO attendance (emp_id, att_date, check_in, check_out, status)
            VALUES (%s, CURDATE(), %s, %s, %s)
            ON DUPLICATE KEY UPDATE
                check_in = VALUES(check_in),
                check_out = VALUES(check_out),
                status = VALUES(status)
        ''', (emp_id, check_in or None, check_out or None, status))
    mysql.connection.commit()
    cur.close()
    flash('Attendance recorded successfully.', 'success')
    return redirect(url_for('attendance.mark_attendance'))
cur.close()
return render_template('attendance/mark.html',
```

```
                        employees=employees, today=today)


@att_bp.route('/history')
@login_required
def attendance_history():
    emp_id = request.args.get('emp_id', '')
    start = request.args.get('start_date', '')
    end = request.args.get('end_date', '')
    cur = mysql.connection.cursor()
    query = '''
        SELECT e.first_name, e.last_name, d.dept_name,
               a.att_date, a.check_in, a.check_out, a.status
        FROM attendance a
        JOIN employee e ON a.emp_id = e.emp_id
        LEFT JOIN department d ON e.dept_id = d.dept_id
        WHERE 1=1
    '''
    params = []
    if emp_id:
        query += ' AND a.emp_id = %s'
        params.append(emp_id)
    if start:
        query += ' AND a.att_date >= %s'
        params.append(start)
    if end:
```

```
        query += ' AND a.att_date <= %s'
        params.append(end)
    query += ' ORDER BY a.att_date DESC'
    cur.execute(query, params)
    records = cur.fetchall()
    cur.execute('SELECT emp_id, first_name, last_name FROM employee WHERE
status=%s', ('Active',))
    employees = cur.fetchall()
    cur.close()
    return render_template('attendance/history.html',
                           records=records, employees=employees)
```

E7. Leave Management Module (leave/routes.py)

This module enables employees to submit leave requests and allows HR/Admin to review, approve, or reject them. Leave balances are tracked per employee.

```
# leave/routes.py

# Leave request submission and approval management


from flask import (Blueprint, render_template, request,

                   redirect, url_for, flash, session)

from app import mysql

from auth.routes import login_required, admin_required

from datetime import datetime


leave_bp = Blueprint('leave', __name__)


@leave_bp.route('/apply', methods=['GET', 'POST'])

@login_required

def apply_leave():

    if request.method == 'POST':

        emp_id = session['emp_id']

        leave_type = request.form['leave_type']

        start_date = request.form['start_date']

        end_date = request.form['end_date']

        reason = request.form['reason']

        start = datetime.strptime(start_date, '%Y-%m-%d')

        end = datetime.strptime(end_date, '%Y-%m-%d')

        num_days = (end - start).days + 1
```

```
        if num_days <= 0:
            flash('End date must be after start date.', 'danger')
            return redirect(url_for('leave.apply_leave'))
        cur = mysql.connection.cursor()
        cur.execute('''
            INSERT INTO leave_request
            (emp_id, leave_type, start_date, end_date, num_days, reason)
            VALUES (%s, %s, %s, %s, %s, %s)
        ''', (emp_id, leave_type, start_date, end_date, num_days, reason))
        mysql.connection.commit()
        cur.close()
        flash('Leave application submitted successfully.', 'success')
        return redirect(url_for('leave.my_leaves'))
    return render_template('leave/apply.html')


@leave_bp.route('/my-leaves')
@login_required
def my_leaves():
    cur = mysql.connection.cursor()
    cur.execute('''
        SELECT * FROM leave_request
        WHERE emp_id = %s ORDER BY applied_on DESC
    ''', (session['emp_id'],))
    leaves = cur.fetchall()
    cur.close()
```

```
    return render_template('leave/my_leaves.html', leaves=leaves)


@leave_bp.route('/all')
@login_required
@admin_required
def all_leaves():
    cur = mysql.connection.cursor()
    cur.execute('''
        SELECT lr.*, e.first_name, e.last_name, d.dept_name
        FROM leave_request lr
        JOIN employee e ON lr.emp_id = e.emp_id
        LEFT JOIN department d ON e.dept_id = d.dept_id
        ORDER BY lr.applied_on DESC
    ''')
    leaves = cur.fetchall()
    cur.close()
    return render_template('leave/all.html', leaves=leaves)


@leave_bp.route('/review/<int:leave_id>', methods=['POST'])
@login_required
@admin_required
def review_leave(leave_id):
    decision = request.form['decision']
    if decision not in ['Approved', 'Rejected']:
        flash('Invalid decision.', 'danger')
```

```
    return redirect(url_for('leave.all_leaves'))
cur = mysql.connection.cursor()
cur.execute('''
    UPDATE leave_request
    SET status = %s, reviewed_by = %s, reviewed_on = NOW()
    WHERE leave_id = %s
''', (decision, session['user_id'], leave_id))
mysql.connection.commit()
cur.close()
flash(f'Leave request has been {decision.lower()}.', 'success')
return redirect(url_for('leave.all_leaves'))
```

E8. Payroll Processing Module (payroll/routes.py)

This module automates salary computation for employees based on their basic salary, allowances, deductions, and attendance records. It generates payroll records and supports payment status tracking.

```
# payroll/routes.py
# Payroll computation and management

from flask import (Blueprint, render_template, request,
                   redirect, url_for, flash)
from app import mysql
from auth.routes import login_required, admin_required

payroll_bp = Blueprint('payroll', __name__)

def compute_net_salary(basic, housing, transport, other_allow,
                       tax, other_deduct):
    """Compute net salary from components."""
    gross = basic + housing + transport + other_allow
    total_deductions = tax + other_deduct
    net = gross - total_deductions
    return round(net, 2)

@payroll_bp.route('/generate', methods=['GET', 'POST'])
@login_required
@admin_required
def generate_payroll():
```

```
cur = mysql.connection.cursor()
cur.execute('''
    SELECT emp_id, first_name, last_name, salary
    FROM employee WHERE status = 'Active'
''')
employees = cur.fetchall()
if request.method == 'POST':
    pay_period = request.form['pay_period']
    for emp in employees:
        emp_id = emp['emp_id']
        basic = float(request.form.get(f'basic_{emp_id}', emp['salary']))
        housing = float(request.form.get(f'housing_{emp_id}', 0))
        transport = float(request.form.get(f'transport_{emp_id}', 0))
        other_allow = float(request.form.get(f'other_allow_{emp_id}', 0))
        tax = float(request.form.get(f'tax_{emp_id}', 0))
        other_deduct = float(request.form.get(f'other_deduct_{emp_id}', 0))
        net = compute_net_salary(basic, housing, transport,
                                 other_allow, tax, other_deduct)
        cur.execute('''
            INSERT INTO payroll (emp_id, pay_period, basic_salary,
            housing_allowance, transport_allowance, other_allowances,
            tax_deduction, other_deductions, net_salary)
            VALUES (%s, %s, %s, %s, %s, %s, %s, %s, %s)
        ''', (emp_id, pay_period, basic, housing, transport,
              other_allow, tax, other_deduct, net))
```

```
            mysql.connection.commit()
            cur.close()
            flash(f'Payroll for {pay_period} generated successfully.', 'success')
            return redirect(url_for('payroll.payroll_list'))
    cur.close()
    return render_template('payroll/generate.html', employees=employees)


@payroll_bp.route('/list')
@login_required
def payroll_list():
    period = request.args.get('period', '')
    cur = mysql.connection.cursor()
    query = '''
        SELECT p.*, e.first_name, e.last_name, d.dept_name
        FROM payroll p
        JOIN employee e ON p.emp_id = e.emp_id
        LEFT JOIN department d ON e.dept_id = d.dept_id
    '''
    if period:
        query += ' WHERE p.pay_period = %s'
        cur.execute(query + ' ORDER BY e.last_name', (period,))
    else:
        cur.execute(query + ' ORDER BY p.generated_on DESC')
    records = cur.fetchall()
    cur.close()
```

```
    return render_template('payroll/list.html', records=records, period=period)


@payroll_bp.route('/mark-paid/<int:payroll_id>', methods=['POST'])
@login_required
@admin_required
def mark_paid(payroll_id):
    cur = mysql.connection.cursor()
    cur.execute(
        'UPDATE payroll SET payment_status=%s WHERE payroll_id=%s',
        ('Paid', payroll_id)
    )
    mysql.connection.commit()
    cur.close()
    flash('Payment status updated to Paid.', 'success')
    return redirect(url_for('payroll.payroll_list'))
```

E9. Report Generation Module (reports/routes.py)

This module enables HR administrators to generate summarized reports on employees, attendance, leave, and payroll based on selected criteria such as department or date range.

```
# reports/routes.py

# Report generation for personnel management data


from flask import (Blueprint, render_template, request, make_response)

from app import mysql

from auth.routes import login_required, admin_required


report_bp = Blueprint('reports', __name__)


@report_bp.route('/')

@login_required

@admin_required

def report_home():

    return render_template('reports/home.html')


@report_bp.route('/employees')

@login_required

@admin_required

def employee_report():

    dept_id = request.args.get('dept_id', '')

    status = request.args.get('status', 'Active')

    cur = mysql.connection.cursor()

    query = '''
```

```
        SELECT e.emp_id, e.first_name, e.last_name, e.gender,
               e.phone, e.email, e.job_title, e.employment_date,
               e.salary, e.status, d.dept_name
        FROM employee e
        LEFT JOIN department d ON e.dept_id = d.dept_id
        WHERE e.status = %s
    '''
    params = [status]
    if dept_id:
        query += ' AND e.dept_id = %s'
        params.append(dept_id)
    query += ' ORDER BY e.last_name, e.first_name'
    cur.execute(query, params)
    employees = cur.fetchall()
    cur.execute('SELECT * FROM department ORDER BY dept_name')
    departments = cur.fetchall()
    cur.close()
    return render_template('reports/employees.html',
                           employees=employees, departments=departments)


@report_bp.route('/attendance-summary')
@login_required
@admin_required
def attendance_summary():
    start = request.args.get('start_date', '')
```

```
end = request.args.get('end_date', '')
cur = mysql.connection.cursor()
query = '''
    SELECT e.first_name, e.last_name, d.dept_name,
           COUNT(CASE WHEN a.status='Present' THEN 1 END) AS present_days,
           COUNT(CASE WHEN a.status='Absent' THEN 1 END) AS absent_days,
           COUNT(CASE WHEN a.status='Late' THEN 1 END) AS late_days,
           COUNT(a.att_id) AS total_days
    FROM employee e
    LEFT JOIN attendance a ON e.emp_id = a.emp_id
    LEFT JOIN department d ON e.dept_id = d.dept_id
    WHERE e.status = 'Active'
'''
params = []
if start:
    query += ' AND a.att_date >= %s'
    params.append(start)
if end:
    query += ' AND a.att_date <= %s'
    params.append(end)
query += ' GROUP BY e.emp_id ORDER BY e.last_name'
cur.execute(query, params)
summary = cur.fetchall()
cur.close()
return render_template('reports/attendance_summary.html',
```

```
                        summary=summary, start=start, end=end)


@report_bp.route('/payroll-summary')
@login_required
@admin_required
def payroll_summary():
    period = request.args.get('period', '')
    cur = mysql.connection.cursor()
    query = '''
        SELECT e.first_name, e.last_name, d.dept_name,
               p.pay_period, p.basic_salary, p.net_salary,
               p.payment_status
        FROM payroll p
        JOIN employee e ON p.emp_id = e.emp_id
        LEFT JOIN department d ON e.dept_id = d.dept_id
    '''
    if period:
        query += ' WHERE p.pay_period = %s'
        cur.execute(query + ' ORDER BY e.last_name', (period,))
    else:
        cur.execute(query + ' ORDER BY p.generated_on DESC')
    records = cur.fetchall()
    cur.close()
    return render_template('reports/payroll_summary.html',
                           records=records, period=period)
```

E10. Administrative Module — User Management (auth/admin.py)

This module allows system administrators to create, manage, and deactivate user accounts and assign appropriate roles to staff members.

```
# auth/admin.py
# User account and role management for administrators

from flask import (Blueprint, render_template, request,
                   redirect, url_for, flash)
from werkzeug.security import generate_password_hash
from app import mysql
from auth.routes import login_required, admin_required

admin_bp = Blueprint('admin', __name__, url_prefix='/admin')

@admin_bp.route('/users')
@login_required
@admin_required
def list_users():
    cur = mysql.connection.cursor()
    cur.execute('''
        SELECT u.*, e.first_name, e.last_name
        FROM users u
        LEFT JOIN employee e ON u.emp_id = e.emp_id
        ORDER BY u.username
    ''')
    users = cur.fetchall()
```

```
    cur.close()
    return render_template('admin/users.html', users=users)


@admin_bp.route('/users/create', methods=['GET', 'POST'])
@login_required
@admin_required
def create_user():
    cur = mysql.connection.cursor()
    cur.execute('''
        SELECT emp_id, first_name, last_name
        FROM employee WHERE status = 'Active'
    ''')
    employees = cur.fetchall()
    if request.method == 'POST':
        username = request.form['username'].strip()
        password = request.form['password']
        role = request.form['role']
        emp_id = request.form.get('emp_id') or None
        pwd_hash = generate_password_hash(password)
        cur.execute('''
            INSERT INTO users (username, password_hash, emp_id, role)
            VALUES (%s, %s, %s, %s)
        ''', (username, pwd_hash, emp_id, role))
        mysql.connection.commit()
        cur.close()
```

```
        flash('User account created successfully.', 'success')
        return redirect(url_for('admin.list_users'))
    cur.close()
    return render_template('admin/create_user.html', employees=employees)


@admin_bp.route('/users/toggle/<int:user_id>', methods=['POST'])
@login_required
@admin_required
def toggle_user(user_id):
    cur = mysql.connection.cursor()
    cur.execute('SELECT is_active FROM users WHERE user_id = %s', (user_id,))
    user = cur.fetchone()
    new_status = 0 if user['is_active'] else 1
    cur.execute('UPDATE users SET is_active=%s WHERE user_id=%s',
                (new_status, user_id))
    mysql.connection.commit()
    cur.close()
    state = 'activated' if new_status else 'deactivated'
    flash(f'User account {state} successfully.', 'info')
    return redirect(url_for('admin.list_users'))
```

E11. Sample HTML Template — Login Page (templates/auth/login.html)

This Jinja2 HTML template renders the login interface, including form validation feedback using Flask's flash messaging system.

```
<!-- templates/auth/login.html -->
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>PMIS | Login</title>
  <link rel="stylesheet"
      href="{{ url_for('static', filename='css/style.css') }}">
</head>
<body class="login-page">
 <div class="login-container">
  <div class="login-card">
   <h2>Personnel Management Information System</h2>
   <h4>Staff Login</h4>
   {% with messages = get_flashed_messages(with_categories=true) %}
    {% if messages %}
     {% for category, message in messages %}
      <div class="alert alert-{{ category }}">{{ message }}</div>
     {% endfor %}
    {% endif %}
   {% endwith %}
   <form method="POST" action="{{ url_for('auth.login') }}">
```

```
      <div class="form-group">
        <label for="username">Username</label>
        <input type="text" id="username" name="username"
              required placeholder="Enter your username">
      </div>
      <div class="form-group">
        <label for="password">Password</label>
        <input type="password" id="password" name="password"
              required placeholder="Enter your password">
      </div>
      <button type="submit" class="btn-login">Login</button>
    </form>
  </div>
 </div>
</body>
</html>
```

E12. Python Dependencies (requirements.txt)

This file lists all Python packages required to run the PMIS application. They can be installed using: pip install -r requirements.txt

```
# requirements.txt

# PMIS Python Package Dependencies



Flask==2.3.3

Flask-MySQLdb==1.0.1

Werkzeug==2.3.7

mysqlclient==2.2.0

Jinja2==3.1.2

itsdangerous==2.1.2

click==8.1.7

python-dotenv==1.0.0
```