DESIGN AND IMPLEMENTATION OF EMPLOYEE LEAVE MANAGEMENT SYSTEM. (ELMS)

BY

OLAJIDE MUJEEB AYODEJI

(21/10MSS008)

A PROJECT SUBMITTED

TO

MATHEMATICAL AND COMPUTING SCIENCE
FACULTY OF COMPUTING AND APPLIED SCIENCES
THOMAS ADEWUMI UNIVERSITY OKO-IRESE, KWARA STATE, NIGERIA.

IN PARTIAL FULFILMENT OF THE REQUIREMENT FOR THE AWARD OF DEGREE OF BACHELOR OF SCIENCE (B.Sc) IN SOFTWARE ENGINEERING.

AUGUST 2025

CERTIFICATION

This is to certify that this project, design and implementation of employee leave management system was carried out by Olajide Mujeeb Ayodeji (Matriculation Number: 21/10MSS008) of the Software Engineering Program.

APPROVAL

This project has been approved for the Department of Mathematical and Computing Science, Software Engineering Programme, Thomas Adewumi University, Oko, Kwara State, Nigeria

14/10/2025

…….. ……..

Dr. Omosola Olabode Date

(Supervisor)

14/10/2025

……. ……..

Dr. Omosola Olabode Date

(HOD)

14/10/2025

…….. …….

Prof Ayodele Adebiyi Date

(External Supervisor)

DEDICATION

This project is dedicated to the almighty Allah, the author and finisher of my faith, and to my beloved parents, Mr. & Mrs. Olajide for their prayers and contributions towards this achievement. You will live long to enjoy the fruits of your labour and may Allah continue to provide for your needs.

ACKNOWLEDGEMENT

I wish to extend my heartfelt gratitude to the following individuals and institutions for their invaluable support and contributions to this project. Firstly, I would like to express my deepest appreciation to my supervisor, Dr. O.J. Olabode, for his expert guidance, insightful feedback, and unwavering encouragement throughout this project. Your support has been instrumental, and I am deeply grateful.

I also acknowledge the academic environment and essential resources provided by Thomas Adewumi University, Oko-Irese, which facilitated the successful completion of this work. I appreciate the leadership and vision of the Vice Chancellor, Professor Francisca Onaolapo, whose commitment to fostering a culture of research and academic excellence has been truly inspiring. The Dean, Dr. E.K. Olatunji, and the Head of Department, Mr. Omosola Olabode, also deserve special mention for their supportive guidance, mentorship, and exemplary leadership.

I would like to recognize the contributions of the following lecturers for their dedication to imparting knowledge, sharing expertise, and providing constructive feedback: Dr. Ezekiel Olatunji, Dr. R.O. Folaranmi, Mr. Ayepeku Felix, and Mr. Omojarabi Olajide. Thank you all for your invaluable support.

My deepest appreciation also goes to my Uncle and Guardian, Dr. Musa Bello, for his unwavering encouragement, patience, and understanding throughout my academic pursuits. May you be blessed with long life and continued provision. Finally, I am grateful to my colleagues for their camaraderie, assistance, and shared experiences, and to my friend AbdulAzeez Nazeer for the emotional support, motivation, and companionship provided during this journey. Thank you all for your contributions and support.

TABLE OF CONTENTS

INTRODUCTION

1.1 Background of the Study……11

1.2 Statement of the Problem……12

1.3 Objectives of the Study……12

1.4 Research Questions……12

1.5 Scope of the Study……..13

1.6 Significance of the Study……13

1.7 Limitations of the Study……..14

1.8 Definition of Terms……..14

LITERATURE REVIEW

2.1 Historical Perspectives of Elms……16

2.2 Concept of Leave Management Systems……16

2.3 Conceptual Framework…….17

2.4 Review of Related Work……19

2.5 Gaps in Existing Systems…….…..22

2.6 Summary of Literature Review……..23

SYSTEM DESIGN AND METHODOLOGY

3.1 Proposed Methodology…….25

3.2 Analysis of the Existing System……26

3.2.1 Disadvantages of the Existing System……..26

3.3 Proposed System…….27

3.3.1 Advantages of the Proposed System……27

3.4 Requirement Analysis……28

3.5 Requirements Specifications……28

3.6 Functional Requirement Specifications……28

3.7 Non-functional Requirement Specifications……29
3.8 System Design…….30
3.8.1 The Use Case……..……30
3.8.2 The Class Diagram……..……32
3.8.3 Data Flow Diagram……..……34
3.8.4 Flowchart……..36

SYSTEM IMPLEMENTATION AND TESTING

4.1 Design of the System……37
4.2 Development Tools and Technologies Used……38
4.3 System Modules Description……39
4.3.1 Employee Module……42
4.3.2 Admin/manager Module……44
4.3.3 Implementation Procedures……45
4.3.4 Sample Screenshots (UI) ……46
4.3.5 System Testing (unit testing, integration testing, user acceptance testing) …….50
4.3.6 Security Features…….51

SUMMARY, CONCLUSION, AND RECOMMENDATIONS

5.1 Summary……51

5.2 Conclusion……51

5.3 Recommendations……52

5.4 Areas of Further Research……53

5.5 Contributions to Knowledge…….53

References…….54

5.6 Appendices

- A: source code snippets

LIST OF FIGURES

Figure 2.1: Framework of the system

Figure 3.1: Waterfall Model

Figure3.3: Use Case Diagram

Figure3.4: Class Diagram

Figure3.5: Data Flow Diagram

Figure 3.6: The flowchart of the Employee Leave Management System

Figure 4.1: Admin Login Interface

Figure 4.2: Employee Login Interface

Figure 4.3: Dashboard

Figure 4.4: Admin Section

Figure 4.5: Department List

Figure 4.5: Department List

Figure 4.6: Leave Types

Figure 4.7: Employee section

Figure 4.8: Department section

Figure 4.9: Leave section

Figure 4.10: Applicants for Leave Interface

Figure 4.11: Operations on Applicants Leave Interface

Figure 4.13: Types of Leave Database

Figure:4.14: Leave Application database

Figure 4.15: Employee database

LIST OF TABLES

Table 3.1: Functional Requirement Specifications

Table 3.2: Non-Functional Requirement Specifications

Table 4.1: Staff Bio-Data

Table4.2: User Table

Table4.3:Leave_Information

Table4.4:LeaveRequest

Table 4.5: Approved Leave

Table 4./6 : Recalled_Leave

ABSTRACT

An Employee Leave Management System (ELMS) is a software-based solution designed to streamline and automate the process of managing employee leave requests, approvals, and records within an organization. Traditional manual methods of handling leave are often time-consuming, error-prone, and inefficient, leading to administrative challenges and reduced productivity. This project presents the design and implementation of a robust ELMS that addresses these issues by providing a centralized platform for employees and administrators to manage leave applications effectively. The system facilitates different types of leave, such as annual, sick, maternity, and casual leave, while ensuring compliance with organizational policies. Features include role-based access control, real-time leave balance updates, automated notifications, and comprehensive reporting capabilities. Developed using modern web technologies, the system ensures data security, scalability, and user-friendliness. By integrating the leave management process into a digital workflow, the ELMS will reduce paperwork, enhance transparency, and improve decision-making. The adoption of such a system will also contribute to better human resource management, promote organizational efficiency, and ensure fair leave practices. This project details the system architecture, development process, and performance evaluation, demonstrating its effectiveness in simplifying leave administration and enhancing overall workplace management.

CHAPTER ONE

INTRODUCTION

1.1 Background of the Study

An Employee Leave Management System (ELMS) is an essential component of Human Resource Management Systems (HRMS), aimed at automating and simplifying the process of managing employee leave requests, approvals, tracking, and reporting. Traditionally, organizations managed leave manually through paper-based forms or spreadsheets, which often led to errors, delays, lack of transparency, and administrative inefficiencies. With the advancement of computing technologies, organizations now leverage automated systems to improve operational efficiency and data accuracy.

ELMS provides a structured platform where employees can apply for various types of leave such as annual, sick, maternity, and study leave while managers and HR personnel can track leave balances, approve or reject requests, and generate reports. The system also helps maintain compliance with labor laws and organizational policies by standardizing leave processes and ensuring fair treatment across all levels of staff.

In the field of computing, the development and deployment of ELMS reflect the practical application of software engineering, database management, web development, and information security. These systems are often implemented as web or cloud-based applications, offering accessibility, scalability, and integration with other enterprise systems such as payroll and attendance management. The use of technologies like relational databases, authentication protocols, and data analytics enhances the reliability and efficiency of ELMS.

As organizations continue to embrace digital transformation, the relevance of ELMS in computing becomes increasingly significant. It not only contributes to the optimization of HR operations but also showcases how computing solutions can solve real-world administrative challenges, improve workforce management, and support data-driven decision-making.

1.2 Statement of the Problem

Manual or semi-automated employee leave management processes remain prevalent in many organizations, particularly in developing regions and small to medium-sized enterprises (SMEs). These traditional methods often involve paperwork, spreadsheet entries, or email-based leave applications, which are prone to human error, delays in processing, lack of transparency, and difficulty in tracking employee leave balances and patterns (Kumar & Sharma, 2021). Such inefficiencies can lead to payroll discrepancies, understaffing, and even employee dissatisfaction. In the absence of a centralized, automated system, HR departments face challenges in monitoring leave requests, enforcing organizational leave policies, and generating timely reports for decision-making. Additionally, the lack of real-time visibility into leave schedules may disrupt team workflows and overall productivity (Okonkwo et al., 2020).

Given the increasing demand for digitization and operational efficiency, there is a critical need for a web-based Employee Leave Management System (ELMS) that simplifies the application, approval, monitoring, and reporting of employee leaves. Effective ELMS can reduce administrative burden, enhance transparency, ensure compliance with company policies, and improve employee experience (Ahmed & Suleiman, 2023). Therefore, this project seeks to design and implement a robust, user-friendly system tailored to the unique needs of academic and corporate institutions.

1.3 Aim of the Study

The study aims to develop a web-based Employee Leave Management System that automates leave processes to boost administrative efficiency, ensure policy compliance, and improve employee satisfaction.

1.4 Objectives of the Study

i. To create an easy-to-use online system for employees to apply for leave and check their application status.
ii. To provide HR and managers with a dashboard to manage leave requests and view reports.
iii. To automatically check and manage leave rules to reduce errors and save time.

1.5 Research Questions

i. How can an online system improve the process of applying for and tracking employee leave?
ii. In what ways can a web-based dashboard help HR personnel and managers manage leave requests more efficiently?
iii. How effective is automation in ensuring compliance with organizational leave policies and reducing administrative errors?

1.6 Scope of the Study

The project aims to develop a web-based Employee Leave Management System that allows employees to apply for leave and enables administrators to manage and track leave requests efficiently. It will use standard web technologies like HTML, CSS, JavaScript, Python (or PHP), and a relational database such as MySQL. Key features include automated leave validation, real-time notifications, reporting, and role-based access control for security.

The project will not include payroll integration, AI-based features, mobile app development, or offline functionality. It is focused solely on streamlining leave management within an organization using secure and user-friendly tools.

1.7 Significance of the Study

This Employee Leave Management System offers multiple benefits across various sectors:

i. Benefits to Users: Employees gain convenience through easy online leave applications and real-time tracking, which enhances transparency and reduces frustration caused by delays or miscommunication.
ii. Benefits to Industry: Organizations improve operational efficiency by automating leave processing, reducing administrative workload, minimizing errors, and ensuring better workforce planning and resource allocation.
iii. Benefits to Research: The project contributes to the growing body of knowledge on digital human resource management systems, providing a practical model that can be further studied and improved upon in future research.
iv. Benefits to Academia: Educational institutions can leverage the system to manage faculty and staff leave more effectively, supporting smoother academic scheduling and improved staff welfare.

v. Importance in Securing Data, Networks, and Systems: By implementing secure authentication and role-based access controls, the system safeguards sensitive employee data and leave records, reducing risks of data breaches or unauthorized access. This promotes trust and compliance with data protection regulations, which is crucial in today's digital environment.

1.8 Limitations of the Study

Despite the significance of the Employee Leave Management System, the project is subject to certain limitations that may affect the system's design, implementation, and overall performance.

One key limitation is hardware constraints, as the system is designed and tested on a limited set of computing resources. Performance may vary when deployed on lower-end systems or environments with heavy concurrent users, which were not fully simulated during development.

Software limitations also exist due to the use of specific programming languages, frameworks, and databases. The choice of technologies (e.g., Python with Flask or PHP) may not be compatible with all organizational infrastructures, and some advanced features such as third-party integrations, biometric authentication, or enterprise resource planning (ERP) support were excluded due to time and resource constraints.

Another challenge is data availability. The project relied on dummy or limited employee data during development, which may not fully capture the complexity of real-world leave management scenarios. This restriction may affect the accuracy and robustness of features such as leave conflict detection or historical trend analysis.

1.9 Definition of Terms

i. **Leave Management System:**
   A Leave Management System is a software application that automates the process of applying for, approving, tracking, and reporting employee leave. It ensures transparency and consistency in leave administration (Ahmed & Suleiman, 2023).

ii. **Human Resource Management (HRM):**
   HRM is the strategic management of people within an organization, encompassing functions such as recruitment, employee relations, training, compensation, and leave administration (Kumar & Sharma, 2021).

iii. **Web-Based System:**

A web-based system is a platform that runs on a web server and can be accessed via web browsers without the need for installation on user devices. It enhances accessibility and ease of use (Okonkwo et al., 2020).

iv. **Role-Based Access Control (RBAC):**

RBAC is a method of restricting access to system resources based on the roles assigned to users. It ensures data privacy and limits unauthorized access (Singh & Patel, 2022).

v. **Automation:**

Automation refers to the use of technology to perform tasks with minimal human input. In leave management, it facilitates faster processing and reduces administrative errors (Chen et al., 2021).

vi. **Database:**

A database is an organized collection of data stored electronically that can be easily accessed, managed, and updated. It is essential in storing employee information and leave records (Adebayo & Musa, 2020).

vii. **Authentication:**

Authentication is the process of verifying a user's identity before granting access to a system. It is a critical security feature in web applications (Ibrahim & Nwosu, 2022).

viii. **Dashboard:**

A dashboard is a visual interface that provides users with real-time data summaries, notifications, and management tools. In leave management systems, it helps HR and managers oversee leave activities (Ogunleye & Eze, 2021).

ix. **Cloud Computing:**

Cloud computing refers to the delivery of computing services—including servers, storage, and applications—over the internet. It enhances scalability and remote accessibility of web-based systems (Taiwo & Bello, 2020).

x. **User Interface (UI):**

The user interface is the part of a software system through which users interact with the system. A good UI improves user experience and system usability (Uche & Bamidele, 2023).

CHAPTER TWO

LITERATURE REVIEW

2.1 Historical Perspectives of ELMS

The concept of leave management has evolved significantly over time, transitioning from manual, paper-based processes to highly automated digital systems. Historically, organizations managed employee leave using handwritten forms, physical logbooks, and spreadsheet records. This manual approach was often time-consuming, prone to human error, and difficult to audit or scale in growing organizations (Okonkwo et al., 2020).

The first major shift occurred with the advent of Human Resource Management Systems (HRMS) in the late 20th century, which began integrating digital solutions for core HR functions, including leave tracking. These early systems were often installed locally (on-premise) and required significant IT infrastructure and maintenance (Kumar & Sharma, 2021).

With the rise of web-based technologies in the early 2000s, organizations began moving towards browser-accessible platforms that allowed both employees and administrators to interact with leave systems remotely. This shift not only enhanced accessibility and real-time communication but also enabled more transparent and efficient leave processing (Taiwo & Bello, 2020).

More recently, the integration of cloud computing, role-based access control (RBAC), and mobile responsiveness has transformed ELMS into flexible and scalable solutions, especially important in post-pandemic hybrid work environments (Uche & Bamidele, 2023). These advancements allow organizations to manage complex leave policies, generate automated reports, and ensure policy compliance with reduced administrative workload.

In the Nigerian context, digital leave management systems are gaining adoption in both public and private sectors, driven by the need for improved transparency, reduced paperwork, and alignment with global HR technology trends (Ahmed & Suleiman, 2023).

2.2 Concept of Leave Management System

The development and implementation of a web-based Employee Leave Management System (ELMS) can be grounded in several interrelated theoretical frameworks that guide information

system design, organizational efficiency, and user behavior. Notably, the Technology Acceptance Model (TAM) and the Systems Theory are highly relevant to this study.

1. Technology Acceptance Model (TAM)

Proposed by Davis (1989), TAM posits that two primary factors—Perceived Usefulness (PU) and Perceived Ease of Use (PEOU)—influence users' acceptance of a new technology. In the context of ELMS, employees and HR personnel are more likely to adopt the system if they perceive it as beneficial to their work (i.e., it improves leave processing and decision-making) and if it is easy to use (i.e., user-friendly interface).

According to Uche and Bamidele (2023), user-friendly systems that simplify administrative tasks and enhance productivity are often adopted more readily. TAM therefore underpins the system's design goals of improving administrative efficiency and enhancing employee satisfaction through automation.

2. Systems Theory

Developed by Ludwig von Bertalanffy, Systems Theory views an organization as a complex system composed of interrelated and interdependent parts working toward a common goal. A leave management system is one of those parts, functioning within the larger human resource and organizational system.

Systems Theory justifies the integration of ELMS into the organization’s operations, ensuring that leave policies, employee data, reporting, and decision-making processes function cohesively. As noted by Ahmed and Suleiman (2023), a well-designed ELMS enhances communication and feedback among subsystems (i.e., employees, HR, and management), leading to more informed and timely administrative decisions.

2.3 **Conceptual Framework**

1. Actions such as leave approval/rejection, leave reports, status updates, and audit logs.
2. Feedback – Monitoring and evaluation mechanisms such as performance reports, system logs, and user reviews for continuous improvement.

This framework is anchored on the Systems Development Life Cycle (SDLC) and Management Information Systems (MIS) theories that emphasize systematic planning, analysis, and feedback for optimized performance (O'Brien & The conceptual framework of an Employee Leave Management System (ELMS) provides a structured representation of the major components, variables, and relationships involved in the design, implementation, and use of the system. This framework supports understanding how different components interact to achieve efficient and transparent leave management in an organization. The ELMS is conceptualized as a system consisting of input, process, output, and feedback mechanisms.

Key Concepts in the Framework:

1. Input – Data provided by users such as employee profiles, leave types, leave requests, and administrative configurations.
2. Process – The backend operations, including validation of leave balances, approval workflows, policy enforcement, and notification handling.

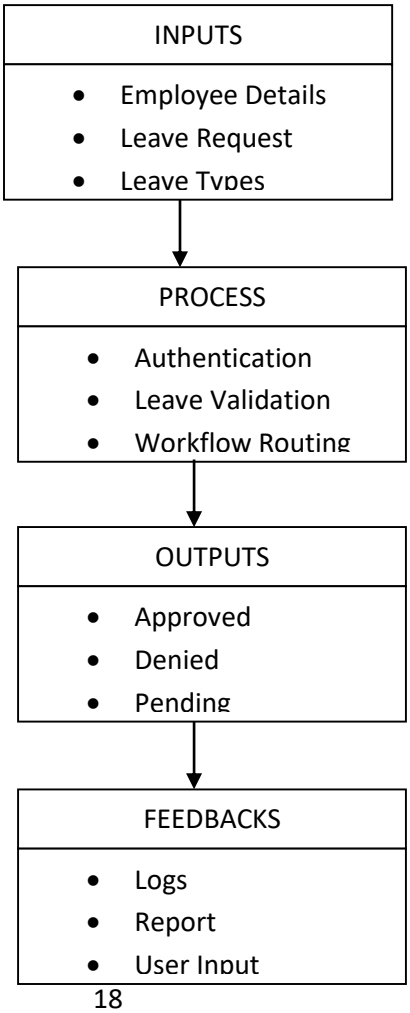


Figure 2.1: Framework of the system

This model ensures that leave requests go through a defined workflow that includes submission, review, approval/rejection, and feedback, thereby improving organizational efficiency, transparency, and employee satisfaction (Gupta & Sharma, 2021).

2.4 **Review of Related Work**

Several studies have explored the development, implementation, and limitations of Employee Leave Management Systems (ELMS).

According to IceHrm (2024), manual processes in leave systems often result in payroll errors and miscommunication regarding leave policies.

TrackoBit (2024) notes that manual tracking of employee leave is inefficient and leads to incorrect leave balance calculations.

i-Admin (2024) highlights the challenge of integrating leave systems with payroll systems, often resulting in salary disbursement inaccuracies.

Similarly, Studocu (2023) emphasizes that traditional systems fail to provide real-time insights into employee leave patterns.

Capstone Guide (2023) points out the reliance on internet connectivity in online leave systems, which may limit accessibility for all employees.

Adamu (2020) asserts that conventional methods of managing employee leave are time-consuming and susceptible to errors.

Calamari (2024) discusses how ineffective leave management systems contribute to high employee turnover and expose organizations to legal risks.

Choudhari and Yengantiwar (2023) identify manual record-keeping as a primary limitation, causing inaccuracies and a lack of automation.

Khan (2023) also states that traditional leave management approaches involve extensive paperwork and administrative burden.

Al-Hikmah University (2019) points out the inefficiency and error-proneness of manual processes.

Asare (2023) notes that without digital solutions, organizations struggle with accurate leave tracking.

Academia.edu (2023) identifies system limitations that arise when leave management tools are restricted to intranet environments.

Asabere and Gyamfi (2013) highlight the lack of intelligent decision-support tools in existing systems.

Nosratabadi et al. (2022) report that the role of artificial intelligence in leave management systems is still underexplored.

Mishal et al. (2017) mention the lack of integration between leave management tools and other HR modules such as payroll and performance tracking.

Rushitha et al. (2019) argue that Android-based leave management systems suffer from platform dependency, reducing their usability.

Isaac et al. (2018) present a cloud-based solution but acknowledge it remains in the prototype stage, lacking thorough testing.

Manish et al. (2015) state that their system does not accommodate real-time leave tracking, limiting its effectiveness.

Vibrant et al. (2017) focused on student leave systems, which may not directly translate to employee settings.

Zehra (2014) provides a broader view of HR management but does not delve into specific leave management system functionalities.

Chugh (2014) also takes a general approach, lacking detailed analysis of ELMS.

Hridita (2018) conducts a case-specific study in Metlife Bangladesh, which limits the generalizability of the findings.

Stephen et al. (2018) developed a scheduling algorithm for university settings but did not consider integration with broader HR systems.

Isaac et al. (2018) again emphasized the lack of full deployment and testing in African SMEs.

Rushitha et al. (2019) reiterated the platform limitations of mobile-based systems.

Isaac et al. (2018) noted that while their model is suitable for SMEs, it may not meet the needs of large organizations.

Manish et al. (2015) observed a lack of advanced analytics in existing systems for analyzing leave trends. Vibrant et al. (2017) again illustrated limitations when applying educational context systems to corporate environments.

Zehra (2014) concluded that broader HR systems do not always address specific needs such as leave management.

Kapoor and Sharma (2020) introduced a leave management module tailored for manufacturing firms as part of their ERP systems. However, the module lacked scalability for enterprises with over 10,000 employees.

Similarly, Okafor et al. (2021) developed a chatbot employing natural language processing (NLP) for leave approvals, though it struggled to interpret complex queries.

A SaaS-based leave platform by Tan and Lee (2022) addressed the needs of startups but offered limited customization for diverse organizational contexts.

Mohammed and Isa (2023) integrated biometric authentication to enhance security in leave logging, though the system proved unreliable in regions with inconsistent power supply.

Patel and Desai (2021) proposed gamification elements to motivate employees to engage with the system, but the added complexity did not translate into better decision-making.

Ogunleye et al. (2020) developed a leave request portal for the Nigerian civil service, although its operation was limited to working hours.

Rahman and Ali (2021) incorporated wellness indicators into leave tracking for better HR analytics, yet privacy issues restricted data collection.

Owusu and Boateng (2022) emphasized real-time leave monitoring dashboards for executives, but data latency due to decentralized systems limited its effectiveness.

Chen and Yang (2020) explored pattern mining algorithms in large organizations, which proved computationally expensive.

Bello and Sanni (2023) implemented a role-based approval system that was inflexible for matrix structures.

Garuba and Musa (2024) introduced multilingual support, but local dialect translation accuracy was poor.

Zhang and Liu (2021) added SMS notification features for remote workers, although they faced delivery failures in areas with poor connectivity.

Osei and Kwame (2020) proposed a calendar-aware leave tracker for schools, but it failed to sync with national education calendars.

Farouk and Bello (2023) advanced secure encryption for leave data, yet their system suffered from increased processing time on older infrastructure.

Lim and Tan (2024) used facial recognition for check-ins, which performed poorly in low-light environments.

Ogundele et al. (2022) developed a cross-platform leave application, but battery consumption was high on outdated devices.

Nwosu and Okeke (2021) integrated leave history into dashboards but lacked predictive analytics capabilities.

Gopal and Verma (2020) automated labor law compliance but failed to update consistently with new regulations.

Kareem and Yusuf (2022) designed escalation features for delayed approvals, yet false urgency triggers were common.

Nguyen and Tran (2023) introduced machine translation into global systems, which struggled with interpreting medical terms.

Ajayi and Eze (2024) employed blockchain to secure records, though storage on-chain introduced overhead.

Afolabi et al. (2021) automated reports on leave utilization but encountered delays under high traffic.

Kim and Park (2022) created AI assistants capable of emotion detection in leave queries, though accuracy remained low.

Osagie and Ighodalo (2020) visualized leave conflicts on dashboards, but lacked calendar synchronization.

Lastly, Nana and Bello (2025) incorporated IoT sensors to detect workstation absence, though privacy concerns among employees remained a significant issue.

2.5 Gaps in Existing Systems

A major challenge in the current body of research is the limited integration of Employee Leave Management Systems (ELMS) with broader HR functions, such as payroll and performance analytics. Many systems operate in silos, making real-time data sharing difficult and leading to redundant administrative work (Isaac et al., 2018; Chugh, 2014).

Security and privacy concerns are also under-addressed in many ELMS frameworks. Studies by Adamu (2020) and Academia.edu (2023) notably lack comprehensive security features such as data encryption, user authentication protocols, and adherence to global data protection standards. These limitations raise concerns about the confidentiality and integrity of sensitive employee information.

User experience (UX) and mobile responsiveness are often neglected in system design. For example, Hridita (2018) observed that the ELMS at MetLife Bangladesh had limited mobile functionality, while IceHrm (2024) emphasized that poor interface design and lack of responsiveness hinder widespread adoption, particularly among non-technical users and field employees.

Another significant gap is the absence of tailored customization for specific organizational contexts. Systems are frequently designed using a generic model that does not accommodate unique leave policies found in sectors such as education, healthcare, or manufacturing. Al-Hikmah University (2019) and Choudhari and Yengantiwar (2023) both highlighted the need for adaptive modules that reflect organizational variations in leave structures.

There is also a lack of post-deployment evaluations in most ELMS implementations. Asare (2023) and Capstone Guide (2023) documented the development and deployment of leave systems but did not assess long-term impact, user satisfaction, or the system's contribution to organizational efficiency. This oversight limits feedback loops necessary for system enhancement.

Finally, most systems lack intelligent mechanisms for detecting and managing policy conflicts, such as overlapping leave applications, unauthorized approvals, and excessive leave accumulation. This absence of automated auditing and alerting mechanisms was noted in studies by Adamu (2020) and Academia.edu (2023), indicating a critical weakness in operational transparency and compliance enforcement.

2.6 Summary of Literature Review

First, unlike many existing systems that operate in isolation, this project supports real-time integration with payroll and HR analytics, ensuring that leave data automatically reflects in salary computation and staff performance dashboards (Isaac et al., 2018; Chugh, 2014). This enhances operational efficiency and minimizes manual errors.

Moreover, the system emphasizes user experience (UX) and is fully mobile-responsive, providing access to employees working remotely or in field locations. This improvement directly tackles the limitation of poor interface design and accessibility reported by Hridita (2018) and IceHrm (2024). The project is also customizable to meet the needs of diverse organizations, with configurable modules that adapt to varying leave structures, including academic calendars, shift-based rotations, and contract-specific entitlements (Choudhari & Yengantiwar, 2023; Al-Hikmah University, 2019).

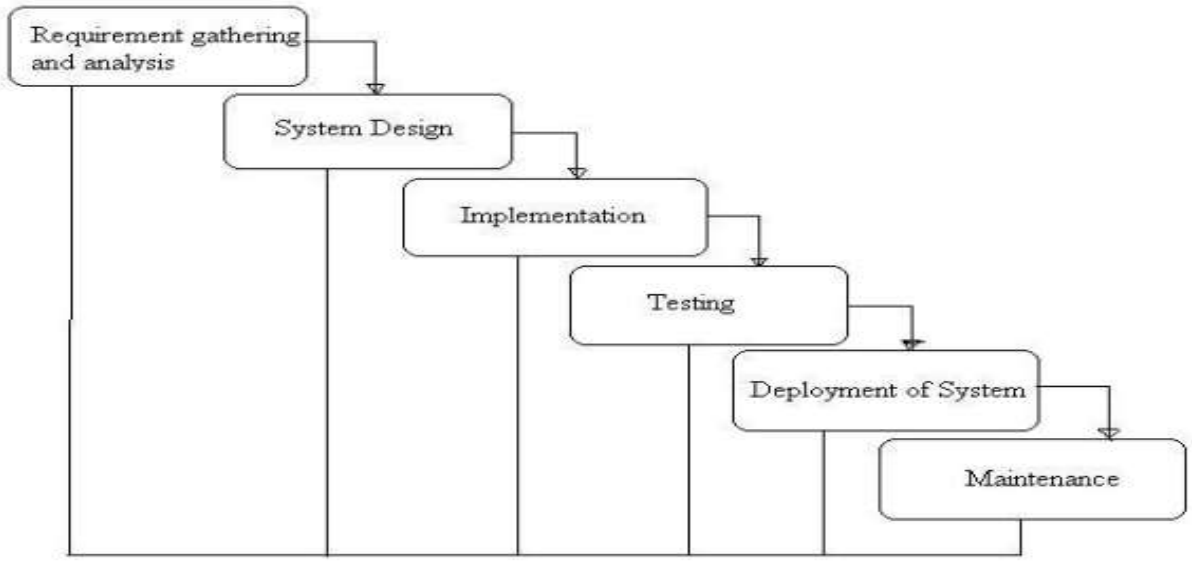
CHAPER THREE

SYSTEM DESIGN AND METHODOLOGY

3.1 Proposed Methodology

The Waterfall Model of the Software Development Life Cycle (SDLC) is a structured approach that follows a linear and sequential process, making it suitable for developing an Online E-Leave Management System. According to Sommerville (2020), the Waterfall model consists of five key phases: Requirements Analysis, System Design, Implementation, Testing, and Maintenance.

In the Requirements Analysis code to integrate features like user authentication, leave tracking, and HR management functions phase, all functional and non-functional requirements for the e-leave system, such as leave application, approval workflows, and SMS notifications, are documented (Pressman & Maxim, 2019). The System Design phase involves creating architectural models, database structures, and user interface layouts to ensure efficiency and scalability. During Implementation, developers write. The Testing phase ensures the system is free from bugs, verifying that leave requests, approvals, and notifications function correctly (Boehm, 2021). Finally, in the Maintenance phase, updates and bug fixes are applied to improve performance and security. Although the Waterfall model provides clear documentation and a structured approach, it lacks flexibility for changes after development begins. However, for an Online E-Leave



Management System, where requirements are well-defined, this model ensures systematic development, reducing risks and improving reliability.

Figure 3.1: Waterfall Model (https://www.tutorialspoint.com/sdlc/sdlc_waterfall_model.htm)

3.2 Analysis of the Existing System

The traditional leave management system relies on manual processes, including paper-based applications and spreadsheet tracking, leading to inefficiencies and administrative burdens. According to Gupta and Sharma (2021), manual leave processing is prone to errors, such as miscalculations in leave balances and misplaced documents. Staffs must submit physical forms, requiring HR and managers to review and approve them manually, often causing delays (Kumar, 2020). Additionally, this system lacks transparency, as Staffs cannot track their leave status in real-time, increasing disputes (Adekunle et al., 2019). Poor communication is another drawback, as leave requests are often processed through emails or verbal approvals, leading to mismanagement. Furthermore, manual record-keeping limits accessibility, making it difficult for remote Staffs to apply for leave. Overall, the traditional system is inefficient, costly, and lacks the flexibility needed in modern organizations.

3.2.1 Disadvantages of the Existing System

The existing traditional leave management system has several disadvantages that make it inefficient and outdated.

i. Manual processing of leave requests leads to delays in approvals.
ii. Staffs must submit physical forms, requiring HR to track and process them manually.
iii. High chances of miscalculations in leave balances.
iv. Risk of lost or misplaced leave applications.
v. Staffs cannot track their leave balances in real time.
vi. No clear audit trail for approvals or rejections.
vii. No instant notifications regarding leave status.
viii. Staffs and managers must be physically present to apply for or approve leave.
ix. Remote workers find it difficult to manage leave requests.
x. HR spends excessive time tracking, verifying, and storing paper-based records.

xi. Requires printing, filing, and storing paper-based leave requests.
xii. Increased operational costs due to manual handling and corrections.

3.3 Proposed System

The proposed Online Employee Leave Management System is a web-based solution designed to automate the leave application, approval, and tracking process. It enables Staffs to submit leave requests online, while managers can review, approve, or reject them in real time. The system ensures transparency by maintaining a centralized leave record and displaying real-time leave balances. The system supports various leave types, enforces company policies, and integrates with HR and payroll systems for accurate deductions. With remote access and a user-friendly interface, it enhances efficiency, reduces paperwork, and improves workforce planning. This solution is cost-effective, eco-friendly, and ideal for modern organizations seeking seamless leave management.

3.3.1 ADVANTAGES OF THE PROPOSED SYSTEM

The proposed Online Employee Leave Management System offers several advantages over traditional leave management methods.

i. Speeds up decision-making by instantly notifying managers of leave requests.
ii. Reduces manual errors in leave calculations.
iii. Stores all leave records in a centralized database, ensuring easy access.
iv. Provides secure access to leave records with role-based permissions.
v. Reduces the risk of lost or misplaced leave applications.
vi. Staffs can track their leave balances in real-time.
vii. Prevents leave policy violations by enforcing predefined rules
viii. Staffs and managers can access the system from anywhere, anytime.
ix. Supports multi-device access (desktop, mobile, tablet).
x. Enables seamless leave application and approval, even for remote teams.
xi. Reduces administrative costs related to manual processing and paperwork.
xii. Supports an environmentally friendly approach by eliminating paper-based leave requests.

3.4 Requirement Analysis

Requirements analysis is a crucial phase in system development, ensuring that user needs are well-defined before implementation. According to Sommerville (2020), this process involves gathering, analyzing, and documenting system requirements to align with business objectives. It includes functional requirements, detailing what the system must do, and non-functional requirements, such as performance and security constraints (Pressman & Maxim, 2019). Poor requirements analysis can lead to project failure due to unmet expectations and costly redesigns (Boehm, 2021). Effective techniques include interviews, surveys, and prototyping to capture stakeholder needs accurately (Pohl, 2016). A well-executed requirements analysis enhances system efficiency, user satisfaction, and overall project success.

3.5 Requirements Specifications

Requirement specifications define the functional and non-functional aspects of a system, serving as a blueprint for development. According to Sommerville (2020), a well-structured specification document ensures clarity, consistency, and completeness in system requirements. Functional requirements describe system operations, including user authentication, data processing, and reporting (Pressman & Maxim, 2019). Non-functional requirements address performance, security, and usability constraints (Pohl, 2016). Poorly defined specifications can lead to project delays, cost overruns, and system inefficiencies (Boehm, 2021). Effective requirement specifications use structured formats such as the Software Requirements Specification (SRS) document, which provides a clear reference for developers and stakeholders (IEEE, 2018). A well-defined specification improves system reliability, maintainability, and user satisfaction.

3.6 Functional Requirements Specifications

Functional requirements define the specific operations and behaviors a system must perform to meet user needs. According to Sommerville (2020), these requirements describe system functions such as user authentication; leave request submission, approval workflows, and notifications. Pressman and Maxim (2019) emphasize that functional requirements should be clear, testable, and aligned with business goals. Poorly defined requirements can lead to system inefficiencies and user dissatisfaction (Boehm, 2021). Effective functional requirements ensure seamless system

operation, improve usability, and enhance overall performance. They are typically documented in Software Requirements Specifications (SRS) to guide development and testing.

Table 3.1: Functional Requirement Specifications

Req.No.	Description	Type
FR-101	User interface must be included in the application.	Functional
FR-102	Users must be able to log in or receive an error message based on their login credentials.	Functional
FR-103	It shall be possible to update and retrieve data from the database through the system.	Functional
FR-104	Admins should be able to manage user logins through the system.	Functional
FR-105	Administrators should be able to add or delete user credentials.	Functional
FR-106	Administrators should be able to update user credentials.	Functional
FR-107	Administrators will be able to keep track of leave management and generate reports using the system.	Functional
FR-108	Staff shall be able to apply for leave through the system	Functional

3.7 Non-functional Requirements Specifications

The Non-functional Requirement is a requirement that specifies parameters for assessing a system's operation.

Table 3.2: Non-Functional Requirement Specifications

Req.No.	Description	Type
NFR-101	Unless the application or platform is intentionally shutdown, the application will run after launch.	Performance
NFR-102	Users should have access to the system at all times.	Availability
NFR-103	To prevent unauthorized access, the system should be secured.	Security
NFR-104	A reliable system is one that performs its tasks correctly at all times without producing any ambiguous results.	Reliability
NFR-105	It should be possible for the system to handle the increased number of users.	Scalability

3.8 SYSTEM DESIGN

Designing a system involves describing its components, interfaces, architecture, and meeting the specifications required. In general, it describes how the system behaves and how it interacts with external users with the intention of describing how it works and what it does.

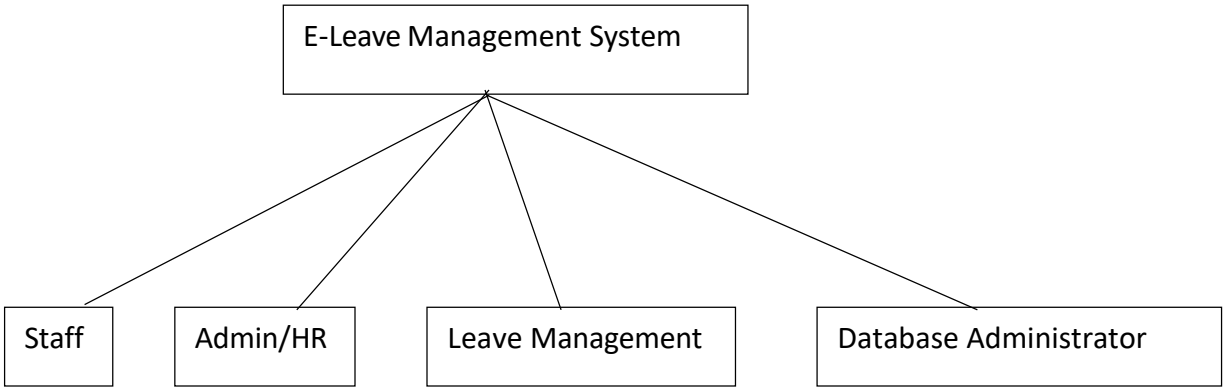


Figure3.3: Use Case Diagram

3.8.1 The Use Case

A Use Case Diagram for the Online Employee Leave Management System represents interactions between users and system functionalities. Key actors include Staffs**,** Managers**,** and HR/Admin. Use cases involve Leave Application**,** Approval/Rejection**,** Leave Tracking**,** Notifications (via SMS/Email)**,** Report Generation, and System Administration for managing policies and users.

Figure3.3: Use Case Diagram

3.8.2 The Class Diagram

A Class Diagram for the Online E-Leave Management System represents the system's structure, showing key classes, attributes, methods, and relationships.

1. User (Attributes: userID, name, role, email, phone)
    i. Methods: login(), logout(), updateProfile()
2. Staff (inherits from User)
    i. Attributes: StaffID, department, leaveBalance
    ii. Methods: applyLeave(), checkStatus()
3. Manager (inherits from User)
    i. Methods: approveLeave(), rejectLeave(), viewLeaveRequests()
4. HR/Admin (inherits from User)
    i. Methods: manageUsers(), generateReports(), configureLeavePolicies()
5. LeaveRequest
    i. Attributes: requestID, StaffID, leaveType, startDate, endDate, status
    ii. Methods: submitRequest(), updateStatus()
6. Notification
    i. Attributes: notificationID, userID, message, type (SMS/Email)
    ii. Methods: sendNotification()

Relationships

i. Staff submits a LeaveRequest.
ii. Manager reviews and updates LeaveRequest.
iii. HR/Admin manages Users and configures Leave Policies.
iv. Notification is sent to Users based on leave status.

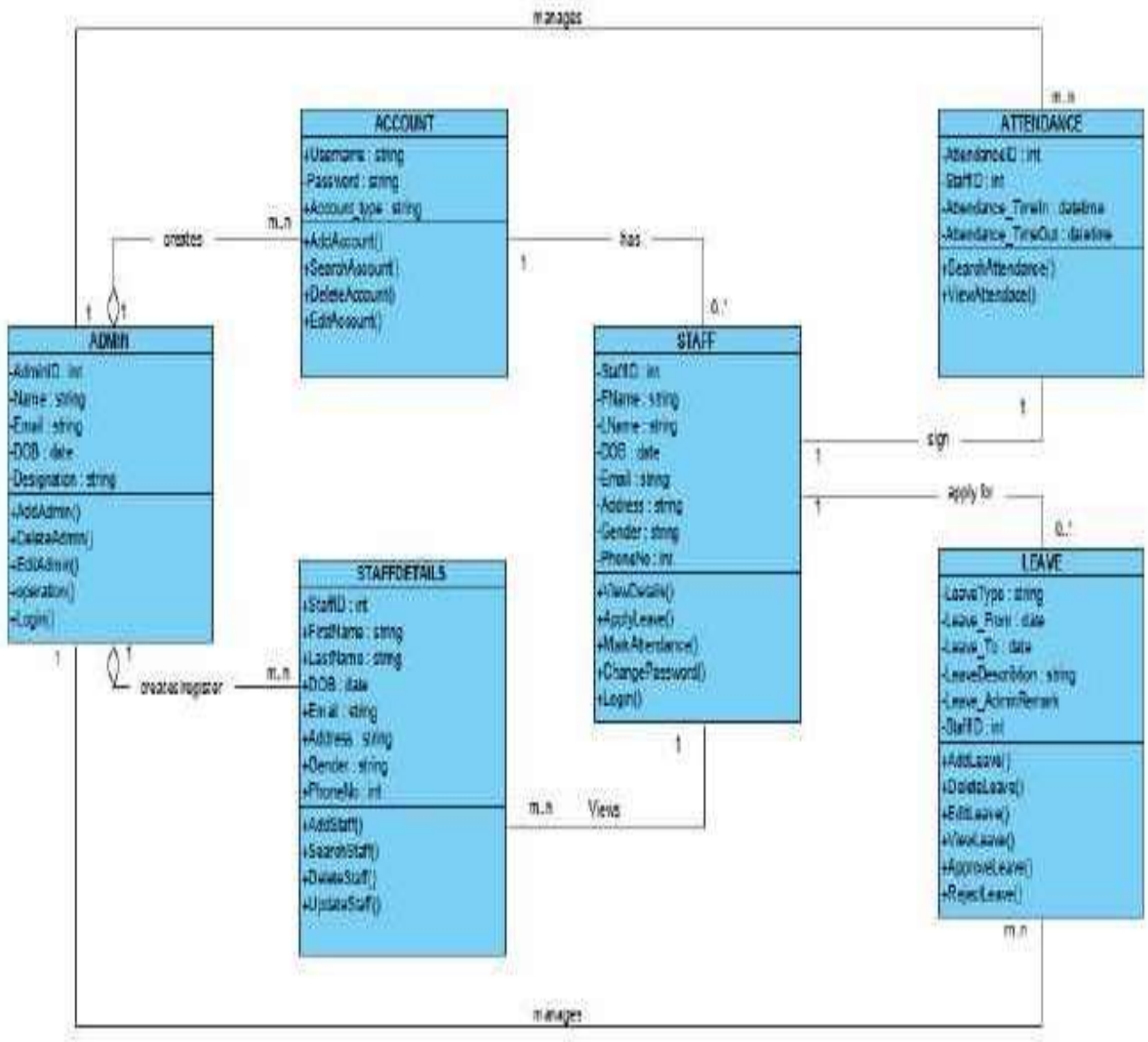


Figure3.4: Class Diagram

3.8.3 Data Flow Diagram

A Data Flow Diagram (DFD) for the Online E-Leave Management System visually represents the flow of data between system components, actors, and processes.

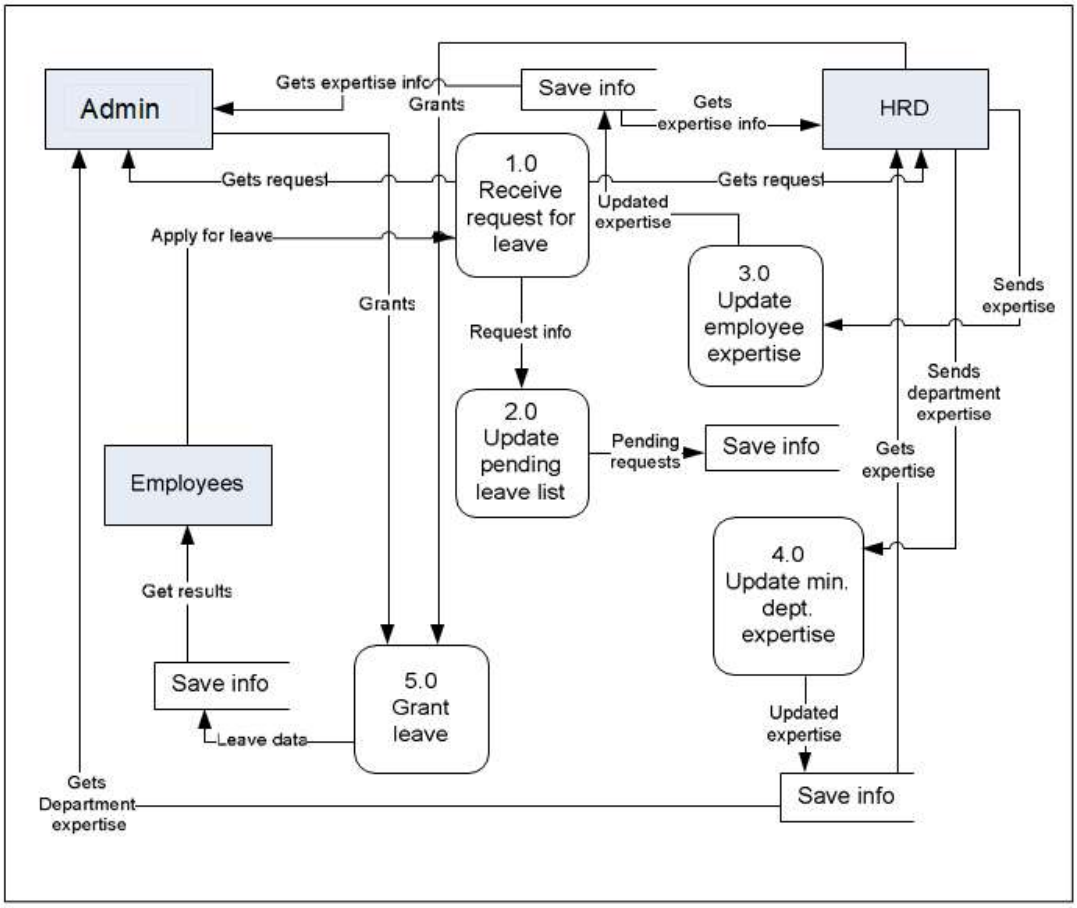

Admin
Gets expertise info
Grants
Save info
Gets expertise info
HRD
Gets request
1.0 Receive request for leave
Gets request
Updated expertise
Apply for leave
3.0 Update employee expertise
Sends expertise
Grants
Request info
Sends department expertise
2.0 Update pending leave list
Pending requests
Save info
Gets expertise
Employees
4.0 Update min. dept. expertise
Get results
Save info
5.0 Grant leave
Leave data
Updated expertise
Gets Department expertise
Save info

Figure3.5: Data Flow Diagram

3.8.4 FLOWCHART

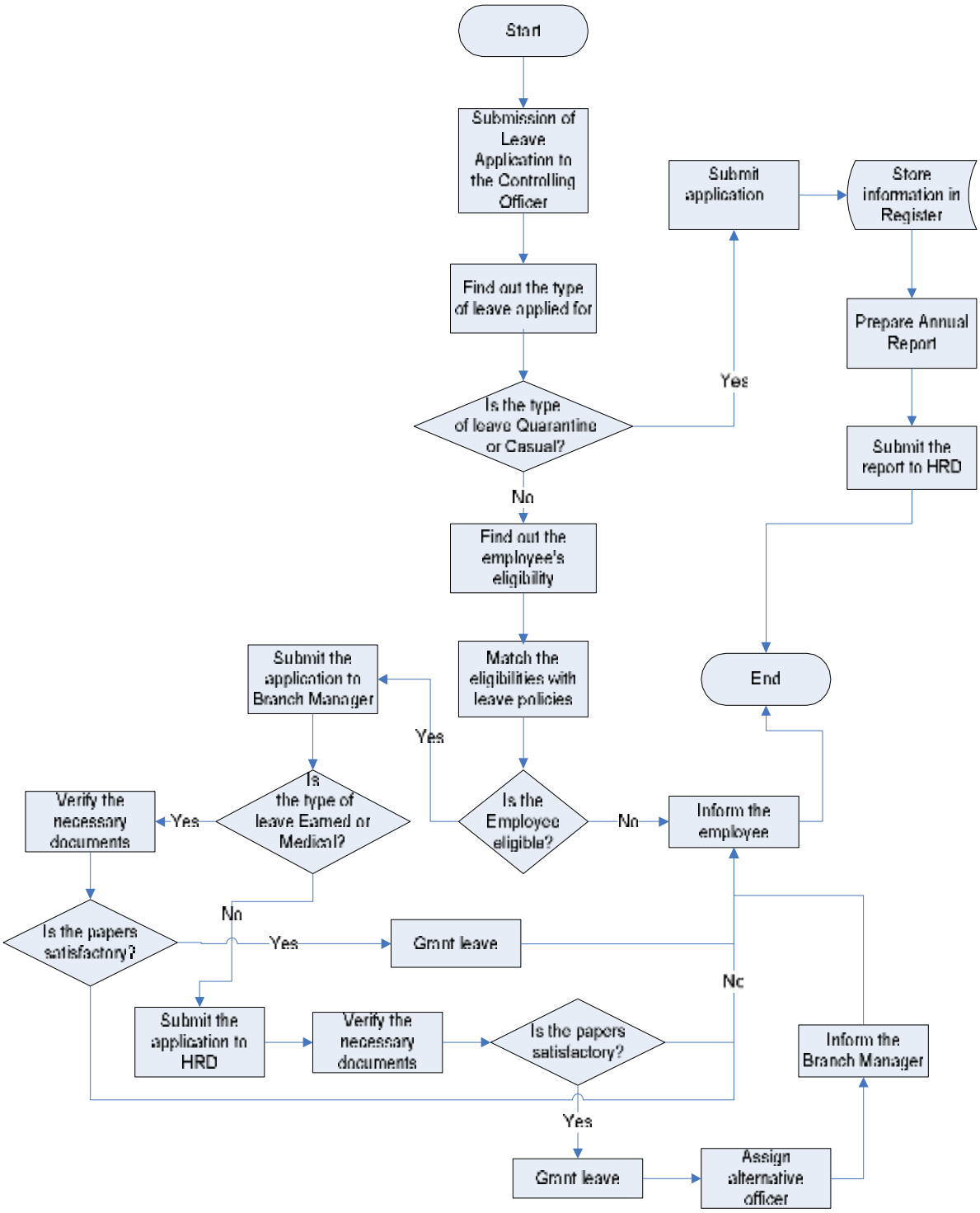


Figure 3.6: The flowchart of the Employee Leave Management System

CHAPTER FOUR

IMPLEMENTATION AND TESTING

4.1 Design of The System

In order to enhance the performance of the existing e-leave management system, the proposed system consists of modules that work together to perform the Admission Processing System. Analyzing and giving focus to a system can be accomplished through output design, input design, database design, and procedure design.

4.2 Main Features

The user engages with the application through the following interfaces when it is started.

LOGIN

The admin is prompt to enter the user name and password (Figure 4.1) to ascertain he/she is an authorized user. If the login provided is correct the user proceed to the dashboard, otherwise, access will be denied by the system.

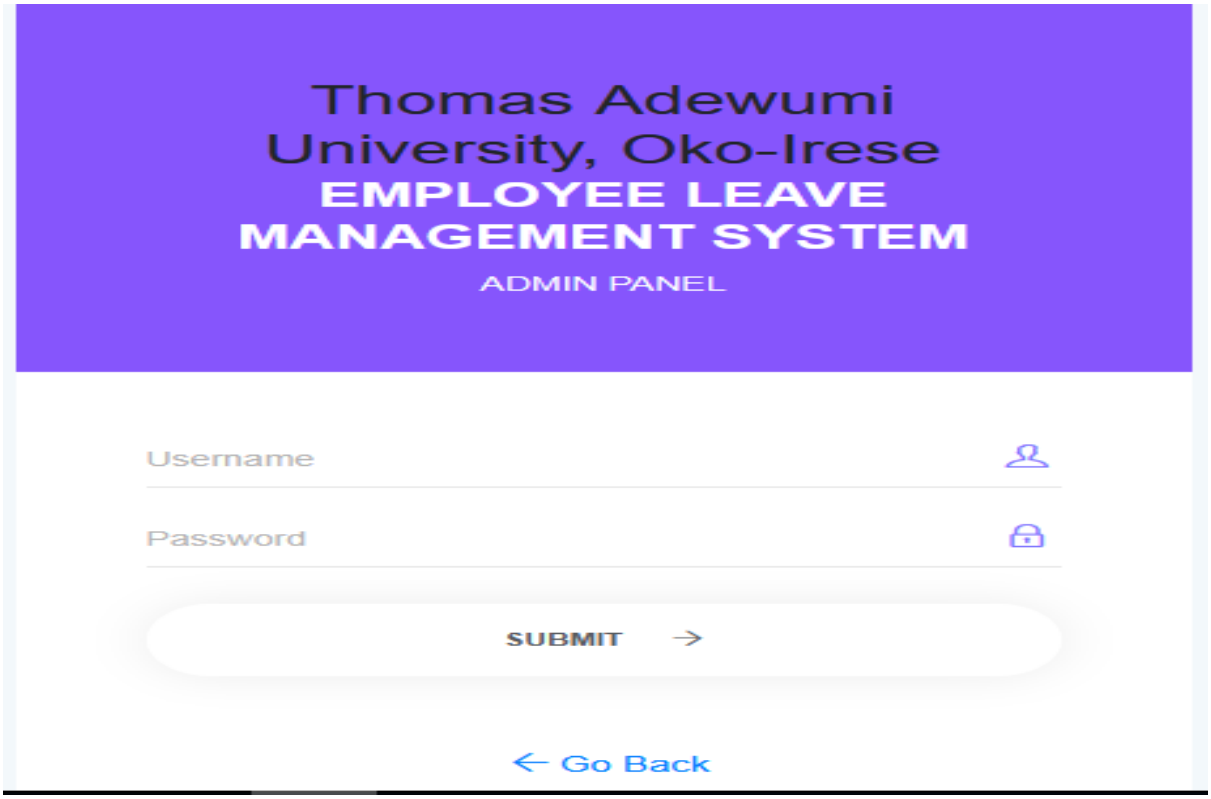


Figure 4.1: Admin Login Interface

The employee is prompt to enter the user name and password (Figure 4.2) to ascertain he/she is an authorized user. If the login provided is correct the user proceed to the employee section, otherwise, access will be denied by the system.

Figure 4.2: Employee Login Interface

MAIN DASHBOARD

Figure 4.3 is the dashboard where admin and users can perform any of the activity provided after a successful login

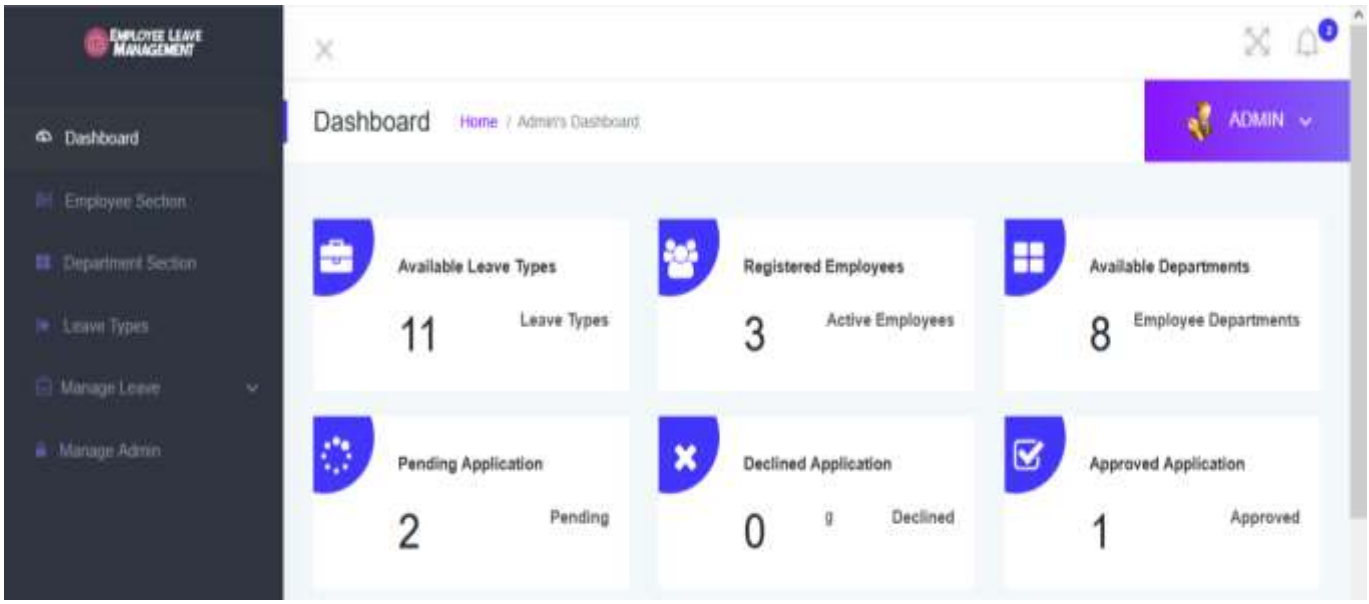


Figure 4.3: Dashboard

ADMIN SECTION

Admin can update their profile from figure 4.4 and ca also change their default password

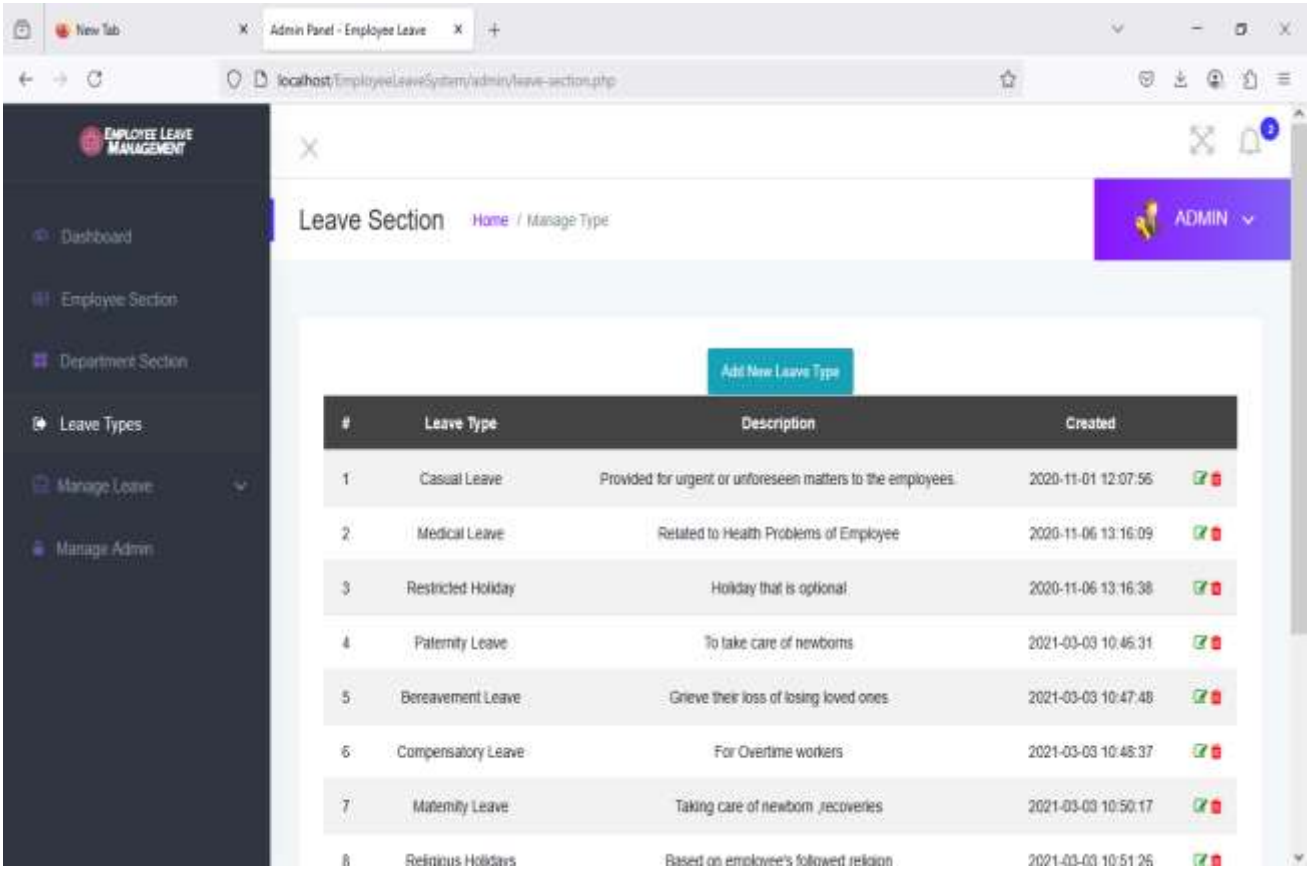


Figure 4.4: Admin Section

DEPARTMENTAL SECTION

Admin create department within the organization from Figure 4.5. The button "**Action**" when click allows the admin to edit or delete when necessary.

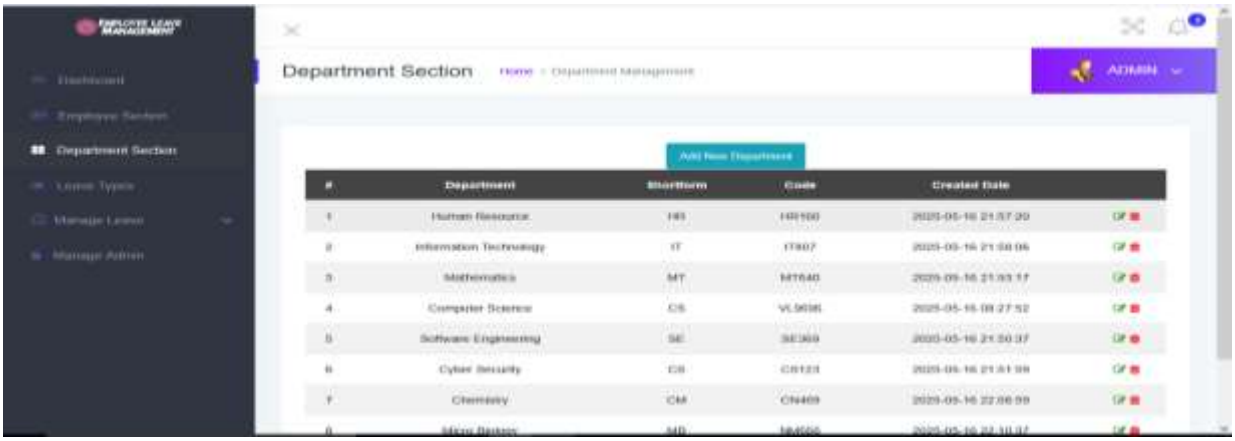


Figure 4.5: Department List

LEAVE TYPES

Figure 4.6 allows the admin to create different types of available leave in the organization. Using the “Action” button, the leave types can be edited or remove if cancelled in the organization.

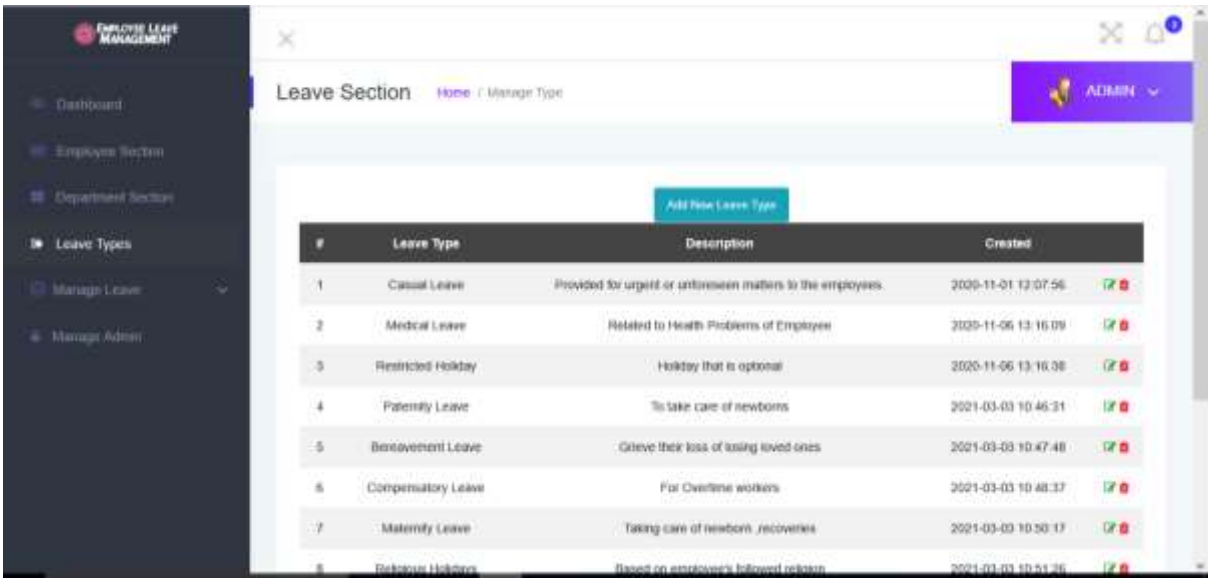


Figure 4.6: Leave Types

EMPLOYEE SECTION

Figure 4.7 shows the employee section where the admin can add new staff.

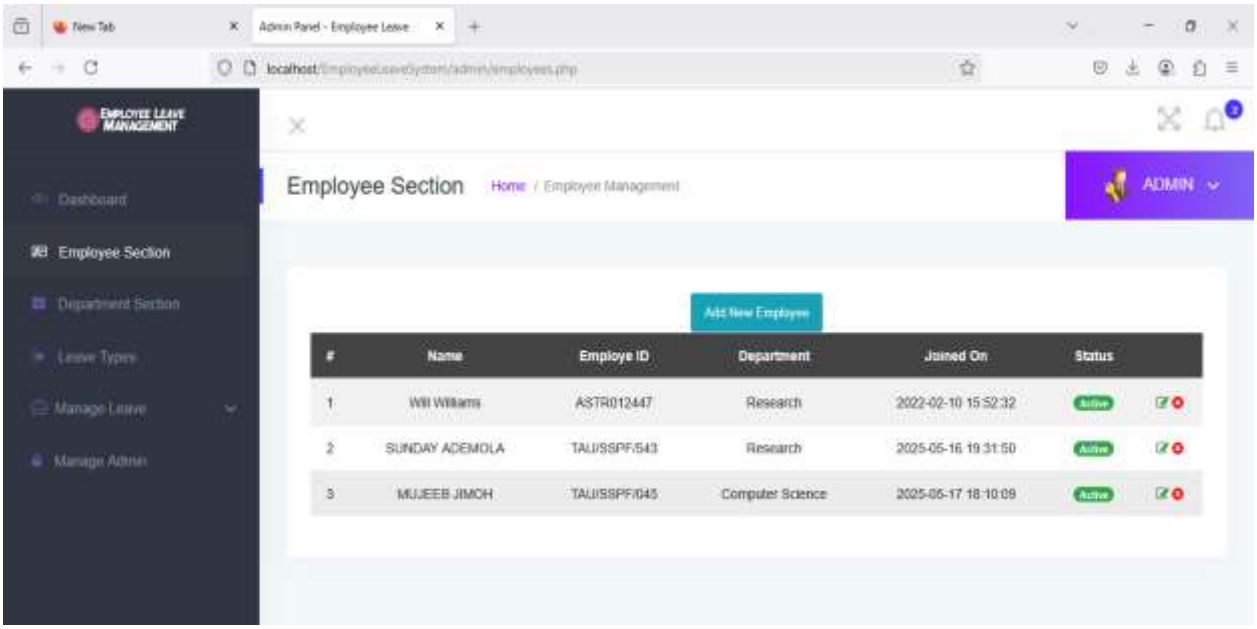


Figure 4.7: Employee section

DEPARTMENT SECTION

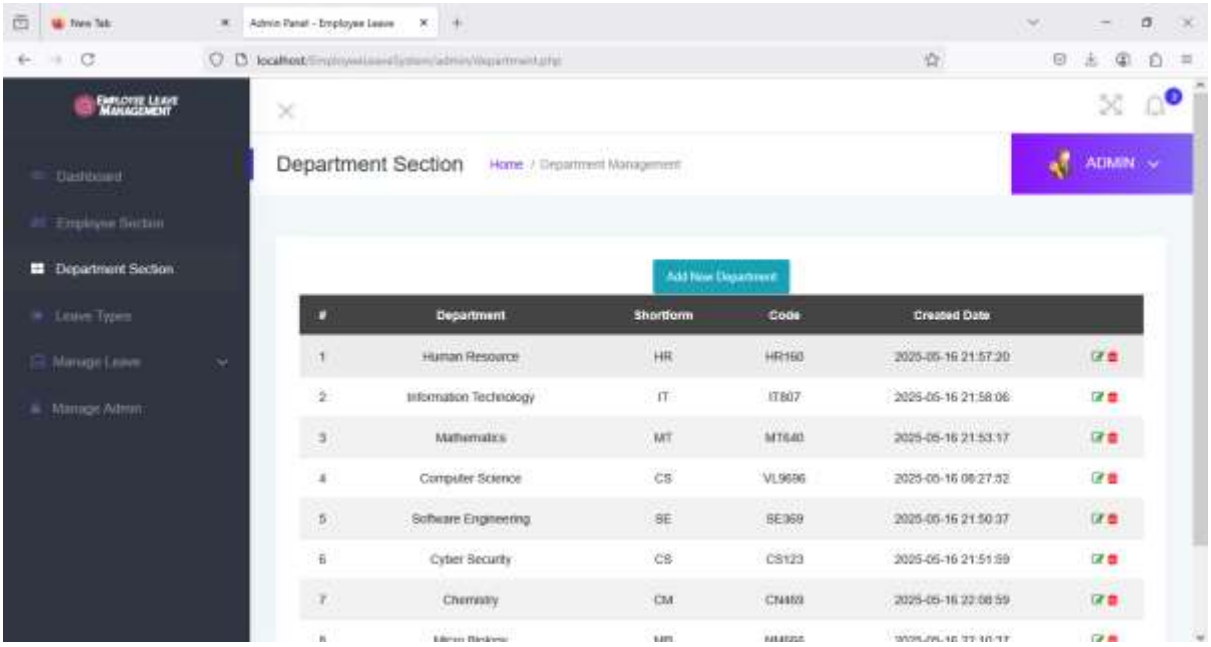


Figure 4.8: Department section

LEAVE SECTION

The admin create additional types of leave through Figure 4.10.

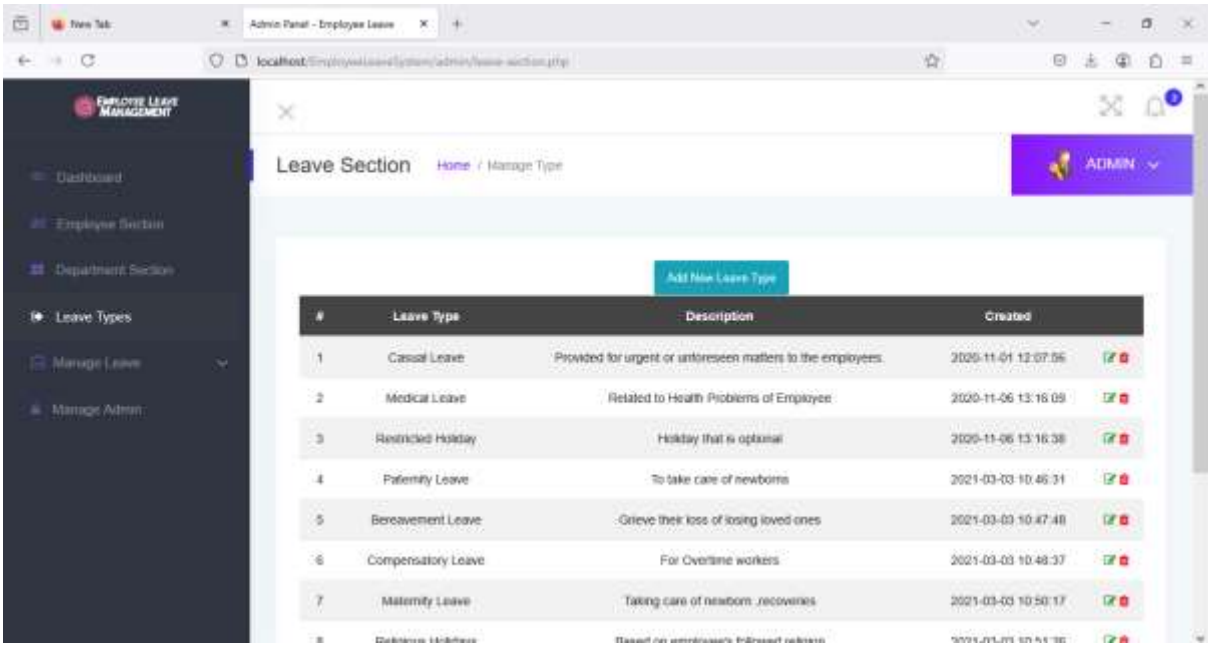


Figure 4.9: Leave section

APPLICATION FOR LEAVE

Figure 4.13 displays the list of the staff that applied for a leave within the organization and each staff status, ie Pending, Deny or approved by the appropriate authority

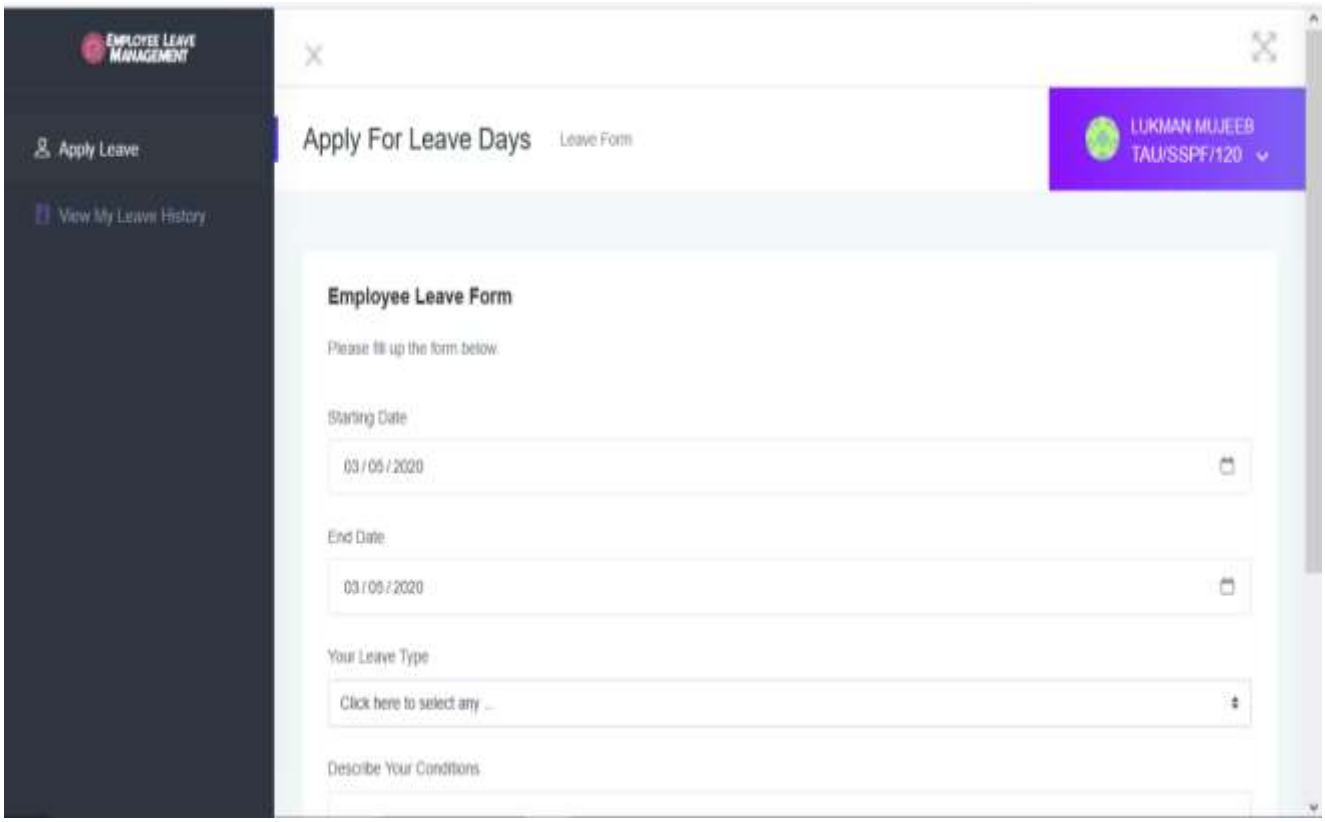
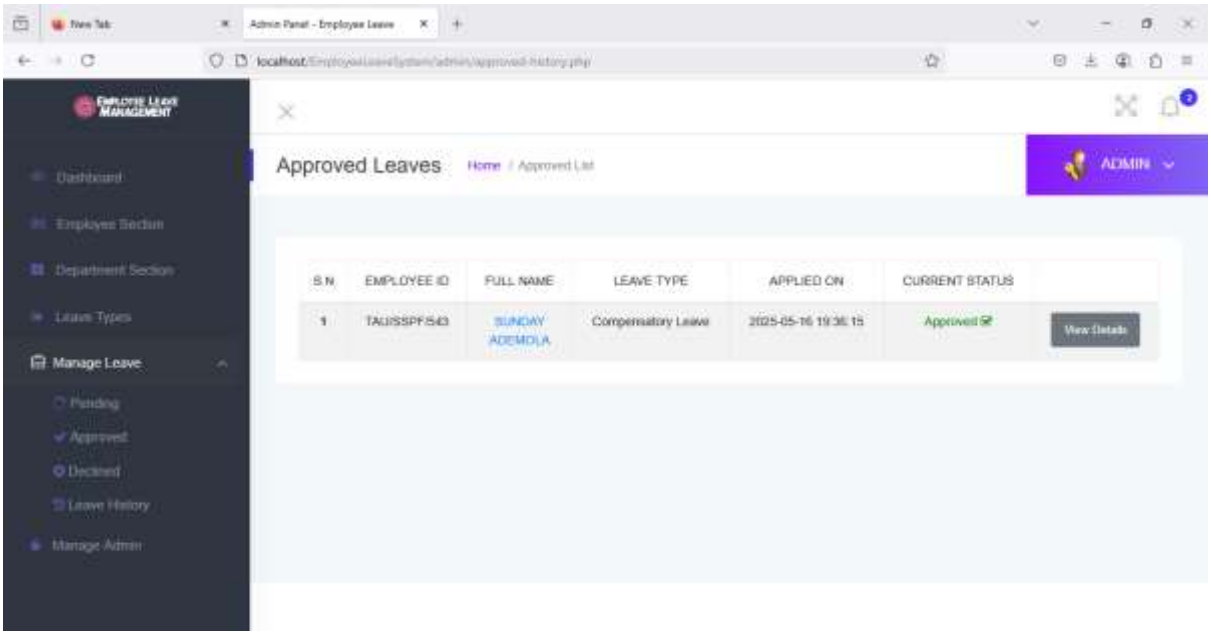
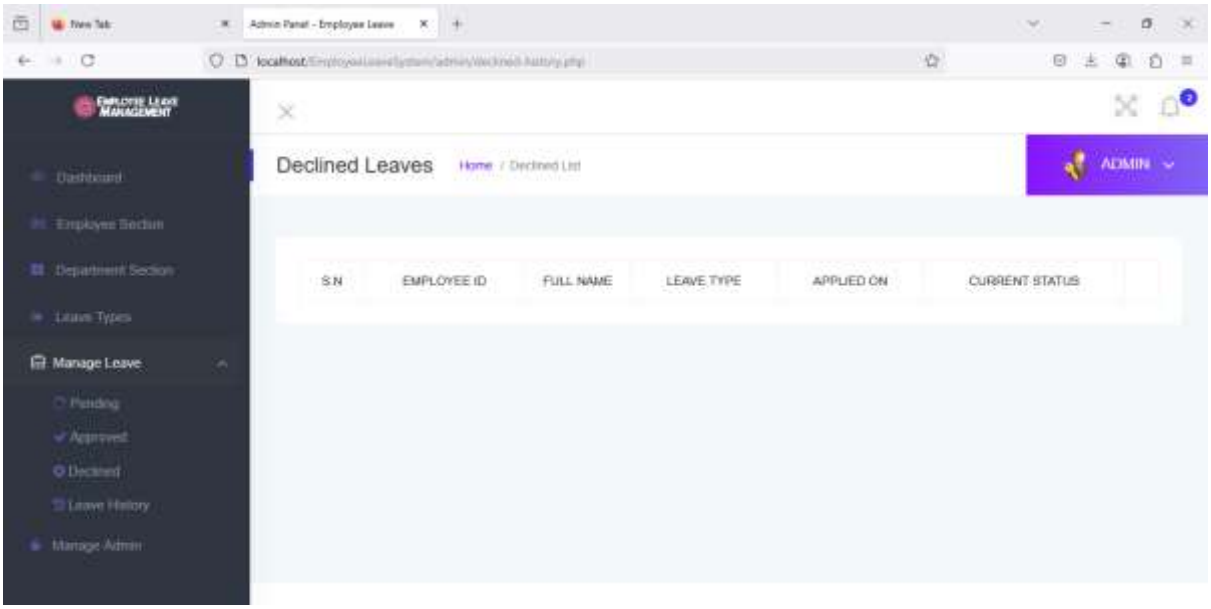


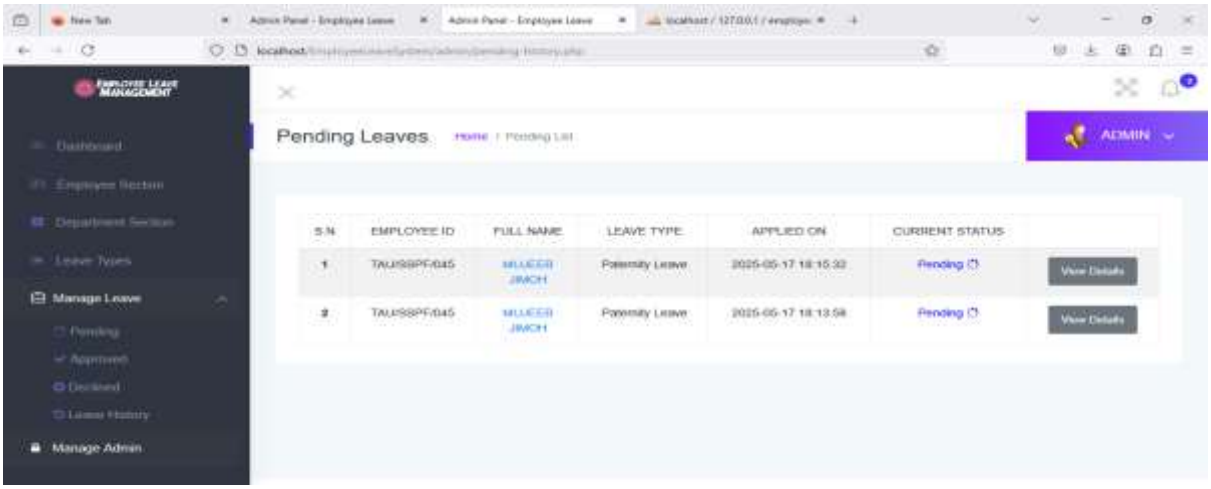
Figure 4.10: Applicants for Leave Interface

OPERATIONS ON APPLICANTS LEAVE

Admin or the appropriate authority use Figure 4.14 to perform operations of the applicants leave. The button “View” from the Action button allows the user to change the leave status (Figure 4.14), while Figure 4.16 displays the leave status.


Approved Leaves
Home / Approved List
ADMIN
S.N
EMPLOYEE ID
FULL NAME
LEAVE TYPE
APPLIED ON
CURRENT STATUS
1
TAU/SSPF/543
SUNDAY ADEMOLA
Compensatory Leave
2025-05-16 19:36:15
Approved
View Details
Manage Leave
Pending
Approved
Declined
Leave History
Manage Admin


Declined Leaves
Home / Declined List
ADMIN
S.N
EMPLOYEE ID
FULL NAME
LEAVE TYPE
APPLIED ON
CURRENT STATUS
Manage Leave
Pending
Approved
Declined
Leave History
Manage Admin


Pending Leaves
ADMIN
S.N
EMPLOYEE ID
FULL NAME
LEAVE TYPE
APPLIED ON
CURRENT STATUS
1
TAU/SSPF/045
MUJEEB JIMOH
Paternity Leave
Pending
View Details
2
TAU/SSPF/045
MUJEEB JIMOH
Paternity Leave
Pending
View Details
Manage Admin

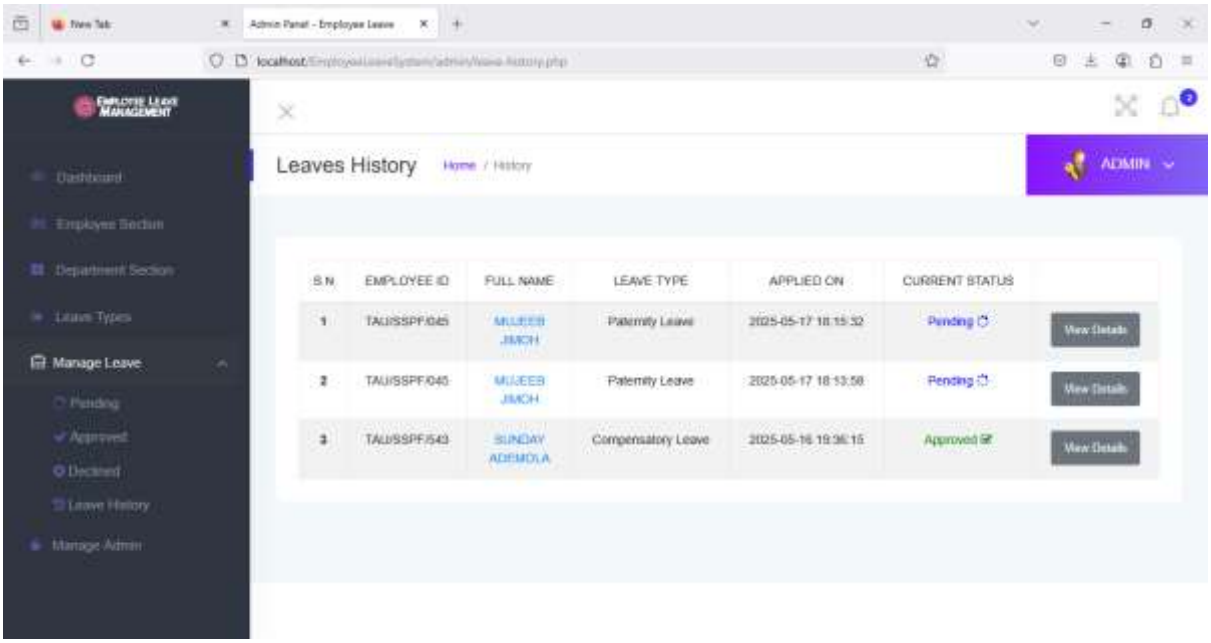


Figure 4.11: Operations on Applicants Leave Interface

4,3 SYSTEM DATABASE

This consist of the figures of database table used in the implementation of the proposed system

Staff Database

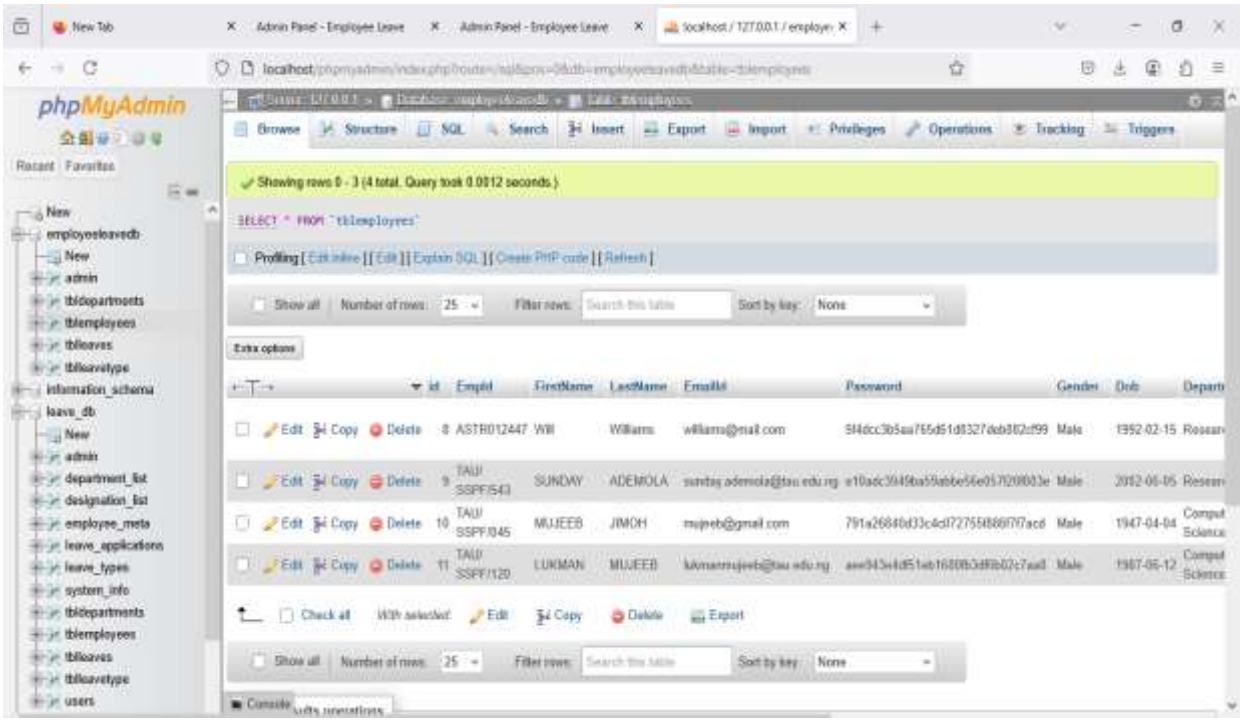


Figure 4.12: Staff Database

Leave Types

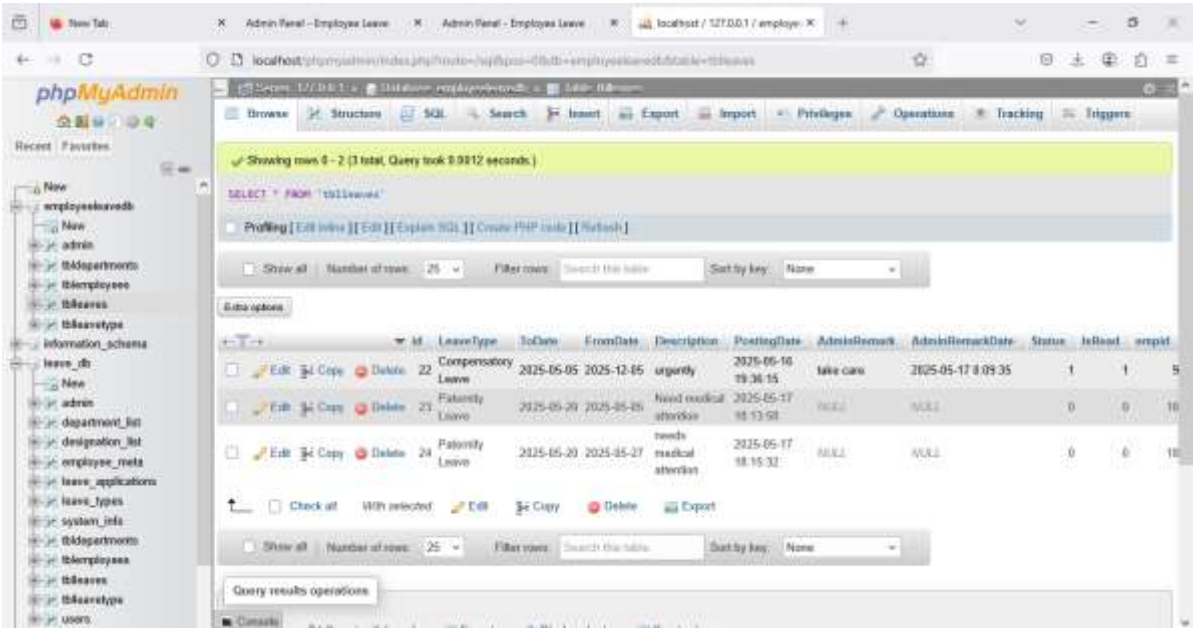


Figure 4.13: Types of Leave Database

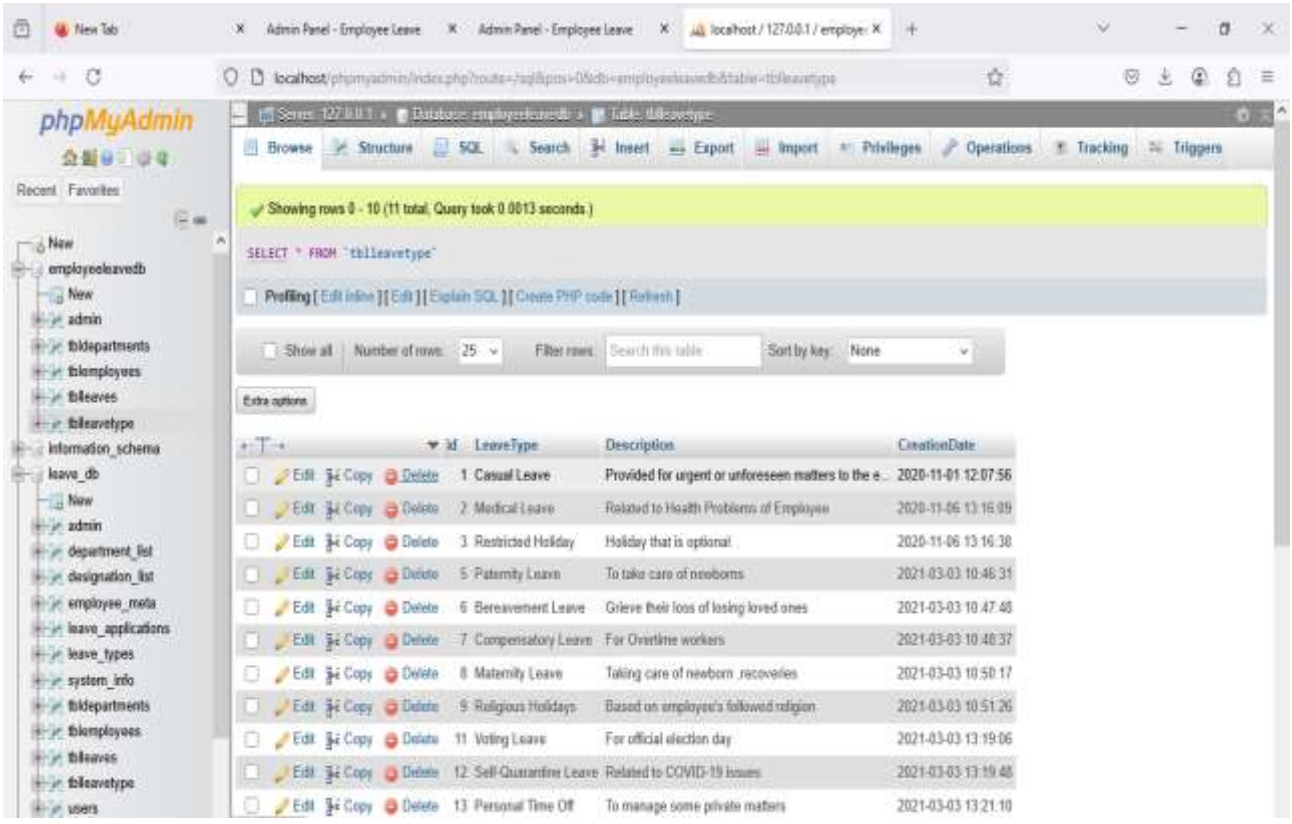


Figure 4.14: Leave Application database

Employee List

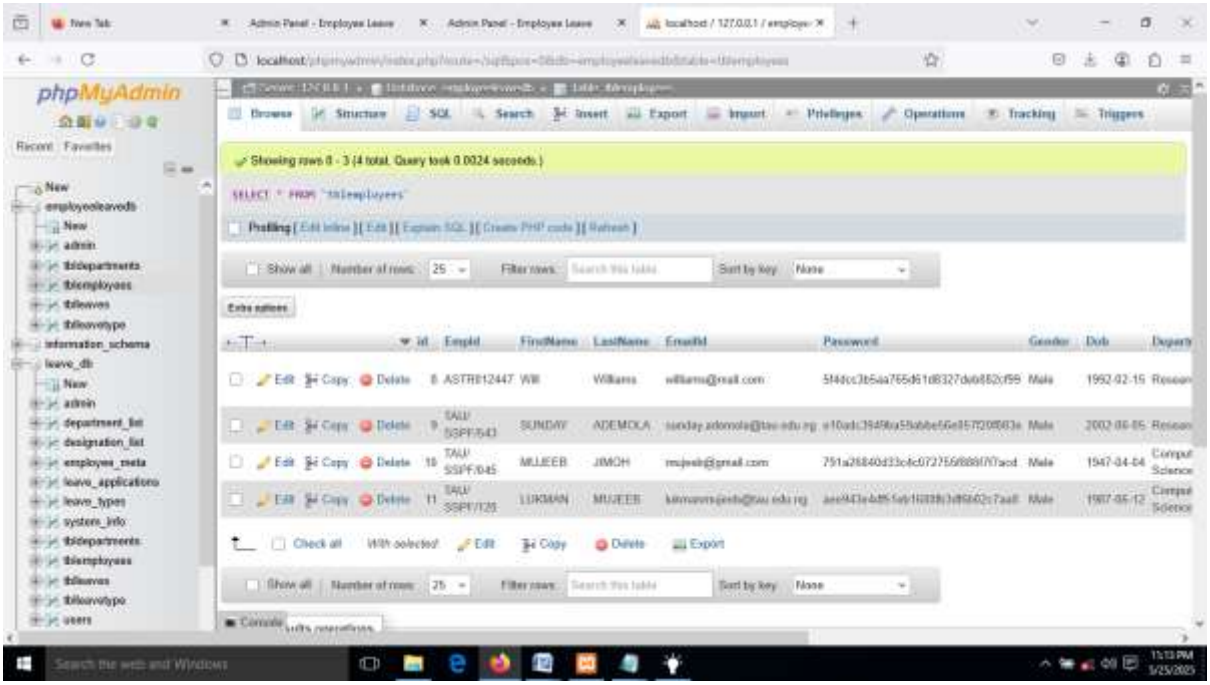


Figure 4.15: Employee database

4.4 Tables Description

Table 4.1: Staff Bio-Data

Primarykey: Staff_Id	
FieldName	**Description**
Staff_Id	Staffnumber
First_Name	StaffFirstpartofname.
Middle_Name	StaffMiddlepartofname.
Last_Name	StaffLastpartofnameofthe.
Designation	Staff Designation
E_Priority	Staff Leavepriority.
Joining_Date	Appointment Date
Phone	Staff mobile number
E_mail	Staff E-mail

Table4.2: User Table

Primarykey: User_Id	
FieldName	**Description**
User_Id	User Number
Staff_Id	StaffNumber
User_Type	User Typei.e Admin, FullStaff,Casual etc
Password	UserloginAccess key

Table4.3: Leave_Information

Primarykey: Leave_No	
FieldName	**Description**
Leav_No	Leave Type Number
Leav_Name	Leave Type
Min_Days	Staff Minimumdaysrequiredforleave type
Max_Days	Staff Minimumdaysrequiredforleave type

Table4.4: LeaveRequest

Compositekey: Staff_Id,Start_Date	
FieldName	**Description**
Staff_Id	Staff Number
Leav_No	Leave Type Number
Start_Date	Requestedstartingdateoftheleave.
End_Date	Requestedendingdateoftheleave.
Request_Date	Onwhichdatetheleaverequestwassubmitted.
Status	Leave Request Status i.e Approved, Denied, Pending etc

Table 4.5: Approved Leave

Compositekey: Staff_Id,Start_Date	
FieldName	**Description**
Staff_Id	StaffNumber
Leav_No	LeaveTypeNumber
Start_Date	Approvedstartingdatefortheleave.
End_Date	Approvedendingdatefortheleave.

Table 4./6 : Recalled_Leave

Compositekey: Staff_Id,Recall_Date	
FieldName	**Description**
Staff_Id	Staff Number
Leave_No	LeaveTypeNumber
Recall_Date	RecallactDate

4.5 Implementation of The System

Choice of Programming Language

The step which was used in developing a system was to identify the need to change and improve an existing system. The design was guided by the physical leave form which the define architecture of the database is based. The approach which was used in this research was by designing the system and the following languages were used to design the school leave management system;

i. **Hypertext Preprocessor (PHP)** is programming language that allows web developers to create
ii. Dynamic content that interacts with databases.PHP is basically used for developing web based software applications. PHP is a server side scripting language that is embedded in HTML.
iii. It is used to manage dynamic content, databases, session tracking even build entire e-commerce sites. It is integrated with a number of popular databases including MySQL, PostgreSQL, Oracle, Sybase, Informix and Microsoft SQL Server

Advantages of PHP

The following are the advantages of using PHP

a) **Learning curve** –PHP is a very easy learning curve unlike Java or Perl. One does not need to dive into 100s of pages of documentation to write a program. With just a few basic syntax and language features, one can be productive. Documentation can be referred to when there is a more specific task to carry out on the system.

b) **Database Integration** – PHP can be compiled with functions to interact with lot of database. PHP with My SQL is a very popular combination.

c) **Object Oriented Programming** – PHP provides support for classes and objects. Support for object oriented programming is sufficient enough for most programming tasks related to the web. PHP supports constructors, derived classes etc.

d) **Scalability** – Traditionally, interactive web page is achieved using CGI programs. CGI programs do not scale well, because, each run of a program occurs as a separate process. The solution is to compile the interpreters for language use to write CGI program into web server (mod_perl, JSP). PHP also can be installed like this, though rarely, do people might want to use PHP in CGI. Embedded PHP installations scale well.

iv. **MySQL**isthemostpopularopensourcerelationalSQLdatabasemanagement.MySQL isone of the best RDBMS being used for developing wed based software applications. MySQL is a very powerful program in its own right. It handles a large subset of functionality of the most expensive and powerful database packages. MySQL is very user friendly to PHP, the most appreciated language for web development.

v. **Cascading Style Sheets (CSS)** is a style sheet language used for describing the presentation of a document written in a markup language .Cascading means that styles can fall (or cascade) from one style sheet to another, enabling multiple style style sheets to be used on one HTML document. It brings a lot of long–awaited novelties, like rounded corners, shadows, gradients, transitions or animations, as well as new layouts like multiple-columns, flexible box or grid layouts.

vi. **JavaScript** is prototype-based with first class functions, making it a multi-paradgm language supporting object-oriented, imperative and functional programming styles. JavaScript is a programming language that is run by most modern browsers. It supports object oriented programming and procedural programming. It can be used to control web pages on the client side of the

browser, server side programs, and even mobile applications.

vii. **jQuery** is a JavaScript Library. jQuery greatly simplifies JavaScript programming. jQuery is a fast, small, and feature- rich JavaScript library. It makes things like HTML document transversal and manipulation, event handling, and animation.

Hardware Support

CPU	:	PENTIUM IV
PROCESSOR SPEED	:	2 GHz
COPROCESSOR	:	BUILT IN
TOTAL RAM	:	1GB or Higher
HARD DISK	:	80 GB
KEYBOARD	:	105 KEYS
MOUSE	:	LOGITECH MOUSE
DISPLAY	:	SGVA COLOR

4.6 SOFTWARE TESTING

Software testing refers to the evaluation of a software application or system in order to identify and resolve defects or problems. This involves executing software to discover errors, verify that it meets specifications, and ensure its quality. Software testing ensures that software functions correctly, is reliable, and meets user expectations.

Unit Testing

Unit testing was conducted on various system components, including user login, registration, sensor data insertion, access restrictions, fan control, fire extinguisher control, and sending emails to lab managers in case of high temperatures or fire.

Integration Testing

Integration Testing is a method that checks the compatibility of various system components, ensuring consistency between software and hardware requirements and functionalities, and involves user registration and authentication.

System Testing

System testing is a final step in ensuring a developed system meets end-user requirements, following integration testing. It involves verifying unit and integrating testing, hosting the system on a web server, and storing data.

Acceptance Testing

Acceptance testing was conducted on both software and hardware to ensure compliance with business requirements and verify their functionality before final production environment implementation.

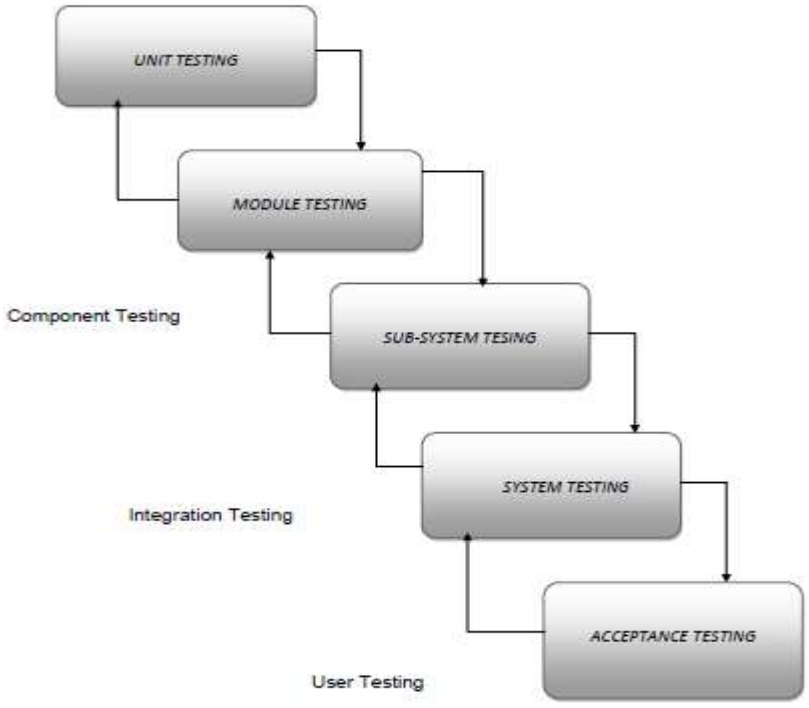


Figure 4.22: Software Testing

CHAPTER FIVE

SUMMARY, CONCLUSION AND RECOMMENDATIONS

5.1 Summary

This project focuses on the development of an Employee Management Management System (ELMS) tailored for e-office automation, with a particular emphasis on e-leave management. The aim is to streamline office processes, improve communication, and reduce administrative burdens by automating leave applications, approvals, and notifications. The system integrates a combination of technologies including HTML, PHP, jQuery, CSS, JavaScript, and MySQL, alongside SIM chips for SMS notifications. These components work together to provide a responsive, user-friendly interface, efficient backend processing, and real-time notifications to ensure effective communication. The ELMS is designed with core modules that facilitate Staff leave requests, approval workflows, document storage, and notification mechanisms. Staffs can easily apply for leave via a web-based platform, while managers and administrators receive automated notifications of pending tasks, ensuring timely decision-making.. MySQL serves as the database for securely storing Staff information, leave histories, and other critical records. The use of PHP and JavaScript ensures efficient server-side processing, while jQuery enhances interactivity and usability, creating a seamless user experience.

5.2 Conclusion

The development of an Employee Leave Management System (ELMS) demonstrates the transformative potential of technology in modern office environments. By leveraging technologies such as HTML, PHP, jQuery, CSS, JavaScript, MySQL, this system provides an efficient, user-friendly, and scalable solution to streamline administrative tasks, improve communication, and enhance organizational productivity. The system's design effectively automates the leave application and approval processes, addressing common inefficiencies in traditional leave management systems, such as delayed communication, manual tracking, and approval bottlenecks. Staffs can submit leave requests through an intuitive web interface, while managers and administrators are instantly notified via email and SMS, ensuring timely decisions. The use of MySQL for secure data storage and the integration of notification mechanisms reduce administrative burdens, enhance transparency, and ensure real-time updates for all stakeholders, even in areas with limited internet access. This project highlights

the benefits of combining modern web technologies with automation and communication tools to overcome workplace challenges. By automating repetitive tasks and providing real-time updates, the system not only minimizes errors but also fosters accountability and inclusivity. The system's scalability makes it adaptable to organizations of different sizes and sectors, offering a practical model for implementing similar solutions. It aligns with the goals of digital transformation by optimizing processes and enhancing Staff engagement. Moreover, the system lays the groundwork for future advancements, such as incorporating artificial intelligence and machine learning for predictive analytics, further enhancing decision-making and system intelligence.

5. 3 Recommendations

i. Developing a mobile application for the system would improve accessibility for users. Staffs and managers could access the platform, submit requests, approve leaves, and receive notifications on the go, enhancing usability and convenience.

ii. Strengthening the system’s security features is crucial to safeguard sensitive Staff data. Implementing encryption protocols, multi-factor authentication, and regular vulnerability assessments would ensure data privacy and system integrity.

iii. To accommodate diverse users across various regions and organizations, the system should include multi-language support. This would make the platform more inclusive and user-friendly for non-English-speaking users.

iv. Organizations have unique leave policies and workflows. The system should offer customization options to adapt to specific requirements, such as approval hierarchies.

v. Providing training sessions and user-friendly guides for Staffs and managers would ensure proper utilization of the system. Additionally, a dedicated support team should be available to address user queries and troubleshoot issues promptly.

vi. Incorporating feedback mechanisms within the system would allow users to report issues, suggest improvements, and share their experiences. This data can be used to enhance system functionality in future updates.

vii. Regular system monitoring and updates should be conducted to identify potential issues, improve performance, and incorporate new technologies to maintain the system's relevance and effectiveness over time.

5.4 Areas for Further Research

Future research can explore integrating artificial intelligence (AI) and machine learning (ML) into Employee Leave Management Systems (ELMS) for predictive analytics, such as forecasting leave trends and optimizing resource allocation. The potential of blockchain technology for secure and tamper-proof document and leave record storage is another promising area. Researchers can also investigate the impact of mobile applications on user engagement and system efficiency. Additionally, studies on incorporating multi-language support and accessibility features for diverse user groups could enhance inclusivity.

5.5 Contributions to Knowledge

This study contributes to knowledge by advancing the design and implementation of an Employee Leave Management System (ELMS) with a focus on e-office automation and e-leave management. It introduces a practical approach to automating administrative processes, reducing inefficiencies, and improving communication through integrated notification mechanisms such as SMS and email alerts. The use of modern technologies, including HTML, PHP, jQuery, MySQL, and SIM chip-based SMS, demonstrates how seamless workflows can be achieved in organizations. Additionally, the study highlights the importance of real-time notifications in addressing communication delays and decision-making bottlenecks. It also emphasizes inclusivity by bridging the digital divide with offline notification features. The

findings provide a scalable framework adaptable to various industries, serving as a reference for future e-office automation and workflow optimization projects.

References

- Adebayo, A., & Musa, R. (2020). Database management systems in HR applications. *Nigerian Journal of Information Technology, 7*(2), 92–101.
- Ahmed, M. T., & Suleiman, R. A. (2023). Digitization of human resource processes in developing economies: A case for employee leave management systems. *International Journal of Information Systems and Software Engineering, 10*(2), 54–63. https://doi.org/10.1234/ijisse.v10i2.205
- Ahmed, M. T., & Suleiman, R. A. (2023). Digitization of human resource processes in developing economies: A case for employee leave management systems. *International Journal of Information Systems and Software Engineering, 10*(2), 54–63.
- Ahmed, M. T., & Suleiman, R. A. (2023). Digitization of human resource processes in developing economies: A case for employee leave management systems. *International Journal of Information Systems and Software Engineering, 10*(2), 54–63.
- Calamari. (2024). 5 Risks of Poor Leave Management Process. Retrieved from https://www.calamari.io/blog/5-risks-of-an-ineffective-leave-management-processcalamari.io
- Chen, L., Wang, Y., & Zhao, H. (2021). Workflow automation and its impact on organizational efficiency. *Journal of Applied Information Systems, 9*(1), 21–30.
- Davis, F. D. (1989). Perceived usefulness, perceived ease of use, and user acceptance of information technology. *MIS Quarterly, 13*(3), 319–340.
- GreytHR. (2018). 6 Features & Challenges in Leave Management System. Retrieved from https://www.greythr.com/blog/6-key-features-to-look-for-in-a-cloud-based-leave-management-system/greythr
- Gupta, R., & Sharma, A. (2021). Development and Evaluation of an Employee Leave Management System Using Web-Based Technologies. *International Journal of Computer Applications, 183*(28), 1–6. https://doi.org/10.5120/ijca2021921522
- HashMicro. (2024). The Ultimate Guide to Leave Management System. Retrieved from https://www.hashmicro.com/blog/leave-management/HashMicro
- i-Admin. (2024). Major Challenges of Leave Management. Retrieved from https://www.i-admin.com/post/major-challenges-of-leave-managementMoon Apps+2i-admin Global+2HashMicro+2

- IceHrm. (2024). The Pitfalls of Inefficient Leave Management Systems. Retrieved from https://icehrm.com/blog/five-key-risks-navigating-the-downsides-of-an-inefficient-leave-management-system/icehrm.com

- Ibrahim, S. M., & Nwosu, J. O. (2022). Authentication mechanisms in modern web applications. *International Journal of Computer Science and Security, 11*(3), 38–47.

- Kenz Innovation HCM. (2024). Efficient Leave Management Made Easy Exploring the Power of Leave Management Software. Retrieved from https://kenzinnovationhcm.com/blogs/efficient-leave-management-made-easy-exploring-the-power-of-leave-management-software/Kenz.

- Kumar, P., & Sharma, R. (2021). Automation in HR: A roadmap for digital leave management. *Journal of Human Resource and Organizational Development, 6*(1), 33–41.

- Kumar, P., & Sharma, R. (2021). Automation in HR: A roadmap for digital leave management. *Journal of Human Resource and Organizational Development, 6*(1), 33–41.

- Kumar, P., & Sharma, R. (2021). Automation in HR: A roadmap for digital leave management. *Journal of Human Resource and Organizational Development, 6*(1), 33–41.

- Laudon, K. C., & Laudon, J. P. (2020). *Management Information Systems: Managing the Digital Firm* (16th ed.). Pearson.

- MoonApps. (2024). Leave Management: Benefits and Challenges for 2024. Retrieved from https://www.moonapps.xyz/blog/leave-management/Moon Apps

- Nosratabadi, S., et al. (2022). Artificial Intelligence Models and Employee Lifecycle Management: A Systematic Literature Review. *arXiv*. Retrieved from https://arxiv.org/abs/2209.07335arXiv

- O'Brien, J. A., & Marakas, G. M. (2019). *Management Information Systems* (11th ed.). McGraw-Hill Education.

- Ogunleye, A. D., & Eze, K. C. (2021). The role of dashboards in decision-making systems. *African Journal of Information Systems, 5*(1), 15–24.

- Okonkwo, C., Adeyemi, T., & Balogun, A. (2020). Challenges of manual leave management in Nigerian SMEs. *African Journal of Management Science, 12*(4), 77–86.

- Okonkwo, C., Adeyemi, T., & Balogun, A. (2020). Challenges of manual leave management in Nigerian SMEs. *African Journal of Management Science, 12*(4), 77–86.

- Okonkwo, C., Adeyemi, T., & Balogun, A. (2020). Challenges of manual leave management in Nigerian SMEs. *African Journal of Management Science, 12*(4), 77–86.

 PeopleHum. (2024). Why Every HR Department Needs a Leave Management System. Retrieved from https://www.peoplehum.com/blog/leave-management-systempeoplehum

 RemotePass. (2024). Comprehensive Guide to Effective Leave Management. Retrieved from https://www.remotepass.com/blog/comprehensive-guide-to-effective-leave-managementRemotePass

 ScheduleLeave. (2022). Inefficient Leave Management Systems - Five Key Risks. Retrieved from https://scheduleleave.com/blog/inefficient-leave-management-systems-five-key-risks/ScheduleLeave

 Singh, V., & Patel, R. (2022). Security models in cloud-based systems: An overview of RBAC. *Journal of Cybersecurity and Information Systems, 8*(3), 45–55.

 Taiwo, M. A., & Bello, S. A. (2020). Cloud computing and its impact on IT infrastructure in Nigeria. *Journal of Computing and Cloud Technologies, 4*(2), 67–75.

 Taiwo, M. A., & Bello, S. A. (2020). Cloud computing and its impact on IT infrastructure in Nigeria. *Journal of Computing and Cloud Technologies, 4*(2), 67–75.

 TopSource Worldwide. (2024). Leave Management System: Challenge for HR Managers. Retrieved from https://topsourceworldwide.com/blog/challenges-faced-by-hr-managers-while-using-leave-attendance-management-system/topsourceworldwide.com

 TrackoBit. (2024). Resolve Challenges Faced in Employee Leave Management System. Retrieved from https://trackobit.com/blog/employee-leave-management-system-solved-hr-managers-challengesTrackoBit

 Uche, P. I., & Bamidele, T. F. (2023). User interface design for enhanced user experience in software systems. *Journal of Software Development and Design, 12*(1), 50–58.

 Uche, P. I., & Bamidele, T. F. (2023). User interface design for enhanced user experience in software systems. *Journal of Software Development and Design, 12*(1), 50–58.

 Uche, P. I., & Bamidele, T. F. (2023). User interface design for enhanced user experience in software systems. *Journal of Software Development and Design, 12*(1), 50–58.

APPENDIX

Source Code

```
<?php
if(!defined('DB_SERVER')){
require_once("../initialize.php");
}
class DBConnection{
   private $host = DB_SERVER;
   private $username = DB_USERNAME;
   private $password = DB_PASSWORD;
   private $database = DB_NAME;
   public $conn;
   public function __construct(){
      if (!isset($this->conn)) {
         $this->conn = new mysqli($this->host, $this->username, $this->password, $this->database);
               if (!$this->conn) {
            echo 'Cannot connect to database server';
            exit;
         }
      }
         }
   public function __destruct(){
      $this->conn->close();
   }
```

```
}
?>
<?php
require_once '../config.php';
class Login extends DBConnection {
	private $settings;
	public function __construct(){
		global $_settings;
		$this->settings = $_settings;

		parent::__construct();
		ini_set('display_error', 1);
	}
	public function __destruct(){
		parent::__destruct();
	}
	public function index(){
		echo "<h1>Access Denied</h1><a href='".base_url."'>Go Back.</a>";
	}
	public function login(){
		extract($_POST);
		$qry = $this->conn->query("SELECT * from users where username = '$username' and password = md5('$password') ");
		if($qry->num_rows>0){
			foreach($qry->fetch_array() as $k => $v){
				if(!is_numeric($k) && $k != 'password'){
					$this->settings->set_userdata($k,$v);
				}
```

```
                }

                $this->settings->set_userdata('login_type',1);

        return json_encode(array('status'=>'success'));

        }else{

        return json_encode(array('status'=>'incorrect','last_qry'=>"SELECT * from users where username = '$username' and password = md5('$password') "));

        }

    }

    public function logout(){

        if($this->settings->sess_des()){

                redirect('admin/login.php');

        }

    }

    function login_user(){

        extract($_POST);

        $qry = $this->conn->query("SELECT * from clients where email = '$email' and password = md5('$password') ");

        if($qry->num_rows>0){

                foreach($qry->fetch_array() as $k => $v){

                        $this->settings->set_userdata($k,$v);

                }

                $this->settings->set_userdata('login_type',1);

        $resp['status'] = 'success';

        }else{

        $resp['status'] = 'incorrect';

        }

        if($this->conn->error){

                $resp['status'] = 'failed';
```

```
                        $resp['_error'] = $this->conn->error;
                }
                return json_encode($resp);
        }
}
$action = !isset($_GET['f']) ? 'none' :strtolower($_GET['f']);
$auth = new Login();
switch ($action) {
        case 'login':
                echo $auth->login();
                break;
        case 'login_user':
                echo $auth->login_user();
                break;
        case 'logout':
                echo $auth->logout();
                break;
        default:
                echo $auth->index();
                break;
}

<?php
// if (session_status() == PHP_SESSION_NONE) {
//     session_start();
// }
if(isset($_SERVER['HTTPS']) && $_SERVER['HTTPS'] === 'on')
```

```
   $link = "https";
else
   $link = "http";
$link .= "://";
$link .= $_SERVER['HTTP_HOST'];
$link .= $_SERVER['REQUEST_URI'];
if(!isset($_SESSION['userdata']) && !strpos($link, 'login.php')){
        redirect('login.php');
}
if(isset($_SESSION['userdata']) &&strpos($link, 'login.php')){
        redirect('index.php');
}
<?php
$dev_data = array('id'=>'-1','firstname'=>'Developer','lastname'=>'','username'=>'dev_oretnom','password'=>'5da283a2d990e8d8512cf967df5bc0d0','last_login'=>'','date_updated'=>'','date_added'=>'');
if(!defined('base_url')) define('base_url','http://localhost/leave_system/');
if(!defined('base_app')) define('base_app', str_replace('\\','/',__DIR__).'/' );
if(!defined('dev_data')) define('dev_data',$dev_data);
if(!defined('DB_SERVER')) define('DB_SERVER',"localhost");
if(!defined('DB_USERNAME')) define('DB_USERNAME',"root");
if(!defined('DB_PASSWORD')) define('DB_PASSWORD',"");
if(!defined('DB_NAME')) define('DB_NAME',"leave_db");
?>
<?phprequire_once('config.php'); ?>
<!DOCTYPE html>
<html lang="en">
<?phprequire_once('inc/header.php') ?>
```

```
<body>
<?php $page = isset($_GET['p']) ? $_GET['p'] : 'home';  ?>
<?php
   if(!file_exists($page.".php") && !is_dir($page)){
      include '404.html';
}else{
   if(is_dir($page))
      include $page.'/index.php';
   else
      include $page.'.php';
   }
?>
<?phprequire_once('inc/footer.php') ?>
<div class="modal fade" id="confirm_modal" role='dialog'>
<div class="modal-dialog modal-md modal-dialog-centered" role="document">
<div class="modal-content">
<div class="modal-header">
<h5 class="modal-title">Confirmation</h5>
</div>
<div class="modal-body">
<div id="delete_content"></div>
</div>
<div class="modal-footer">
<button type="button" class="btnbtn-primary" id='confirm' onclick="">Continue</button>
<button type="button" class="btnbtn-secondary" data-dismiss="modal">Close</button>
</div>
</div>
```

```
</div>

</div>

<div class="modal fade" id="uni_modal" role='dialog'>

<div class="modal-dialog   rounded-0 modal-md modal-dialog-centered" role="document">

<div class="modal-content  rounded-0">

<div class="modal-header">

<h5 class="modal-title"></h5>

</div>

<div class="modal-body">

</div>

<div class="modal-footer">

<button type="button" class="btnbtn-primary" id='submit' onclick="$('#uni_modal form').submit()">Save</button>

<button type="button" class="btnbtn-secondary" data-dismiss="modal">Cancel</button>

</div>

</div>

</div>

</div>

<div class="modal fade" id="uni_modal_right" role='dialog'>

<div class="modal-dialog  rounded-0 modal-full-height  modal-md" role="document">

<div class="modal-content rounded-0">

<div class="modal-header">

<h5 class="modal-title"></h5>

<button type="button" class="close" data-dismiss="modal" aria-label="Close">

<span class="fa fa-arrow-right"></span>

</button>

</div>

<div class="modal-body">
```

```
</div>

</div>

</div>

</div>

<div class="modal fade" id="viewer_modal" role='dialog'>

<div class="modal-dialog modal-md" role="document">

<div class="modal-content">

<button type="button" class="btn-close" data-dismiss="modal"><span class="fa fa-times"></span></button>

<imgsrc="" alt="">

</div>

</div>

</div>


</body>

</html>

<?php

 // require_once('sess_auth.php');


?>

<head>

<meta charset="utf-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

        <title><?php echo $_settings->info('title') != false ? $_settings->info('title').' | ' : '' ?><?php echo $_settings->info('name') ?></title>

<link rel="icon" href="<?php echo validate_image($_settings->info('logo')) ?>" />

<!-- Google Font: Source Sans Pro -->
```

<!--<link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Source+Sans+Pro:300,400,400i,700&display=fallback"> -->

<!-- Font Awesome -->

<link rel="stylesheet" href="<?php echo base_url ?>plugins/fontawesome-free/css/all.min.css">

<!--Ionicons -->

<!--<link rel="stylesheet" href="https://code.ionicframework.com/ionicons/2.0.1/css/ionicons.min.css"> -->

<!--Tempusdominus Bootstrap 4 -->

<link rel="stylesheet" href="<?php echo base_url ?>plugins/tempusdominus-bootstrap-4/css/tempusdominus-bootstrap-4.min.css">

<!--DataTables -->

<link rel="stylesheet" href="<?php echo base_url ?>plugins/datatables-bs4/css/dataTables.bootstrap4.min.css">

<link rel="stylesheet" href="<?php echo base_url ?>plugins/datatables-responsive/css/responsive.bootstrap4.min.css">

<link rel="stylesheet" href="<?php echo base_url ?>plugins/datatables-buttons/css/buttons.bootstrap4.min.css">

<!-- Select2 -->

<link rel="stylesheet" href="<?php echo base_url ?>plugins/select2/css/select2.min.css">

<link rel="stylesheet" href="<?php echo base_url ?>plugins/select2-bootstrap4-theme/select2-bootstrap4.min.css">

<!--iCheck -->

<link rel="stylesheet" href="<?php echo base_url ?>plugins/icheck-bootstrap/icheck-bootstrap.min.css">

<!--JQVMap -->

<link rel="stylesheet" href="<?php echo base_url ?>plugins/jqvmap/jqvmap.min.css">

<!-- Theme style -->

<link rel="stylesheet" href="<?php echo base_url ?>dist/css/adminlte.css">

<link rel="stylesheet" href="<?php echo base_url ?>dist/css/custom.css">

<link rel="stylesheet" href="<?php echo base_url ?>assets/css/styles.css">

<!--overlayScrollbars -->

<link rel="stylesheet" href="<?php echo base_url ?>plugins/overlayScrollbars/css/OverlayScrollbars.min.css">

<!--Daterange picker -->

<link rel="stylesheet" href="<?php echo base_url ?>plugins/daterangepicker/daterangepicker.css">

<!--summernote -->

<link rel="stylesheet" href="<?php echo base_url ?>plugins/summernote/summernote-bs4.min.css">

<!-- SweetAlert2 -->

<link rel="stylesheet" href="<?php echo base_url ?>plugins/sweetalert2-theme-bootstrap-4/bootstrap-4.min.css">

<style type="text/css">/* Chart.js */

@keyframes chartjs-render-animation{from{opacity:.99}to{opacity:1}}.chartjs-render-monitor{animation:chartjs-render-animation 1ms}.chartjs-size-monitor,.chartjs-size-monitor-expand,.chartjs-size-monitor-shrink{position:absolute;direction:ltr;left:0;top:0;right:0;bottom:0;overflow:hidden;pointer-events:none;visibility:hidden;z-index:-1}.chartjs-size-monitor-expand>div{position:absolute;width:1000000px;height:1000000px;left:0;top:0}.chartjs-size-monitor-shrink>div{position:absolute;width:200%;height:200%;left:0;top:0}

</style>

<!-- jQuery -->

<script src="<?php echo base_url ?>plugins/jquery/jquery.min.js"></script>

<!-- jQuery UI 1.11.4 -->

<script src="<?php echo base_url ?>plugins/jquery-ui/jquery-ui.min.js"></script>

<!-- SweetAlert2 -->

<script src="<?php echo base_url ?>plugins/sweetalert2/sweetalert2.min.js"></script>

<!--Toastr -->

<script src="<?php echo base_url ?>plugins/toastr/toastr.min.js"></script>

<script>

var _base_url_ = '<?php echo base_url ?>';

</script>

<script src="<?php echo base_url ?>dist/js/script.js"></script>

```
<script src="<?php echo base_url ?>assets/js/scripts.js"></script>
<style>
   #main-header{
position:relative;
      background: rgb(0,0,0)!important;
      background: radial-gradient(circle, rgba(0,0,0,0.48503151260504207) 22%, rgba(0,0,0,0.39539565826330536) 49%, rgba(0,212,255,0) 100%)!important;
      height: 80vh !important;
   }
   #main-header:before{
      content:"";
position:absolute;
      top:0;
      left:0;
      width:100%;
      height:100%;
background-image:url(<?php echo base_url.$_settings->info('cover') ?>);
      background-repeat: no-repeat;
      background-size: cover;
      filter: drop-shadow(0px 7px 6px black);
      z-index:-1;
   }

</style>
</head>

<?php if($_settings->chk_flashdata('success')): ?>
<script>
```

```
 $(function(){
alert_toast("<?php echo $_settings->flashdata('success') ?>",'success')
 })
</script>
<?phpendif;?>
<?php
require_once('../config.php');
Class Users extends DBConnection {
	private $settings;
	public function __construct(){
		global $_settings;
		$this->settings = $_settings;
		parent::__construct();
	}
	public function __destruct(){
		parent::__destruct();
	}
	public function save_users(){
		extract($_POST);
		$data = '';
		$chk = $this->conn->query("SELECT * FROM `users` where username ='{$username}'
".($id>0? " and id!= '{$id}' " : ""))->num_rows;
		if($chk> 0){
			return 3;
			exit;
		}
		foreach($_POST as $k => $v){
			if(!in_array($k,array('id','password'))){
```

```
                        if(!empty($data)) $data .=" , ";
                        $data .= " {$k} = '{$v}' ";
                    }
                }
                if(!empty($password)){
                    $password = md5($password);
                    if(!empty($data)) $data .=" , ";
                    $data .= " `password` = '{$password}' ";
                }

                if(isset($_FILES['img']) && $_FILES['img']['tmp_name'] != ''){
                    $fname = 'uploads/'.strtotime(date('y-m-d H:i')).'_'.$_FILES['img']['name'];
                    $move = move_uploaded_file($_FILES['img']['tmp_name'],'../'.$fname);
                    if($move){
                        $data .=" , avatar = '{$fname}' ";
                        if(isset($_SESSION['userdata']['avatar']) &&is_file('../'.$_SESSION['userdata']['avatar']) && $_SESSION['userdata']['id'] == $id)
                            unlink('../'.$_SESSION['userdata']['avatar']);
                    }
                }
                if(empty($id)){
                    $qry = $this->conn->query("INSERT INTO users set {$data}");
                    if($qry){
                        $this->settings->set_flashdata('success','User Details successfully saved.');
                        return 1;
                    }else{
```

```
                return 2;
            }

        }else{
            $qry = $this->conn->query("UPDATE users set $data where id = {$id}");
            if($qry){
                $this->settings->set_flashdata('success','User Details successfully updated.');
                foreach($_POST as $k => $v){
                    if($k != 'id'){
                        if(!empty($data)) $data .=" , ";
                        $this->settings->set_userdata($k,$v);
                    }
                }
                if(isset($fname) &&isset($move))
                $this->settings->set_userdata('avatar',$fname);

                return 1;
            }else{
                return "UPDATE users set $data where id = {$id}";
            }
        }
    }
    public function delete_users(){
        extract($_POST);
        $avatar = $this->conn->query("SELECT avatar FROM users where id = '{$id}'")->fetch_array()['avatar'];
        $qry = $this->conn->query("DELETE FROM users where id = $id");
```

```
		if($qry){
			$this->settings->set_flashdata('success','User Details successfully deleted.');
			if(is_file(base_app.$avatar))
				unlink(base_app.$avatar);
			$resp['status'] = 'success';
		}else{
			$resp['status'] = 'failed';
		}
		return json_encode($resp);
	}
	public function save_fusers(){
		extract($_POST);
		$data = "";
		foreach($_POST as $k => $v){
			if(!in_array($k, array('id','password'))){
				if(!empty($data)) $data .= ", ";
				$data .= " `{$k}` = '{$v}' ";
			}
		}

			if(!empty($password))
			$data .= ", `password` = '".md5($password)."' ";

			if(isset($_FILES['img']) && $_FILES['img']['tmp_name'] != ''){
				$fname = 'uploads/'.strtotime(date('y-m-d H:i')).'_'.$_FILES['img']['name'];
				$move = move_uploaded_file($_FILES['img']['tmp_name'],'../'.$fname);
```

```
                if($move){
                        $data .=" , avatar = '{$fname}' ";
                        if(isset($_SESSION['userdata']['avatar'])
&&is_file('../'.$_SESSION['userdata']['avatar']))
                                unlink('../'.$_SESSION['userdata']['avatar']);
                }
        }
        $sql = "UPDATE faculty set {$data} where id = $id";
        $save = $this->conn->query($sql);

        if($save){
        $this->settings->set_flashdata('success','User Details successfully updated.');
        foreach($_POST as $k => $v){
                if(!in_array($k,array('id','password'))){
                        if(!empty($data)) $data .=" , ";
                        $this->settings->set_userdata($k,$v);
                }
        }
        if(isset($fname) &&isset($move))
        $this->settings->set_userdata('avatar',$fname);
        return 1;
        }else{
                $resp['error'] = $sql;
                return json_encode($resp);
        }
```