## A REPORT

## ON

**Human Resource Management Software**

***Submitted by,***

**Ms. Sudheshna Shahabadi - 20211CSE0679**

### *Under the guidance of,*

**Mr. Md Ziaur Rahman**

**Assistant Professor**

***in partial fulfillment for the award of the degree of***

**BACHELOR OF TECHNOLOGY**

**IN**

**COMPUTER SCIENCE AND ENGINEERING**

**At**



**PRESIDENCY UNIVERSITY**

**BENGALURU**

**MAY 2025**

**PRESIDENCY UNIVERSITY**

**PRESIDENCY SCHOOL OF COMPUTER SCIENCE AND ENGINEERING**

**CERTIFICATE**

This is to certify that the Internship report **“Human Resource Management Software”** being submitted by “SUDHESHNA SHAHABADI” bearing roll number “20211CSE0679” in partial fulfillment of the requirement for the award of the degree of Bachelor of Technology in Computer Science and Engineering is a bonafide work carried out under my supervision.

|  |  |
| --- | --- |
| **Mr. Md Ziaur Rahman**  Assistant Professor  PSCS  Presidency University | **Dr. Asif Mohammed**  Associate Professor & HOD  PSCS  Presidency University |

|  |  |
| --- | --- |
| **Dr. MYDHILI NAIR**  Associate Dean  PSCS  Presidency University | **Dr. SAMEERUDDIN KHAN**  Pro-Vice Chancellor -Engineering  Dean –PSCS / PSIS  Presidency University |

**PRESIDENCY UNIVERSITY**

**PRESIDENCY SCHOOL OF COMPUTER SCIENCE AND ENGINEERING**

**DECLARATION**

I hereby declare that the work, which is being presented in the report entitled “**Human Resource Management Software”** in partial fulfillment for the award of Degree of **Bachelor of Technology** in **Computer Science and Engineering**, is a record of my own investigations carried under the guidance of **Mr. Md Ziaur Rahman,** **Assistant Professor,** **Presidency School of Computer Science and Engineering, Presidency University, Bengaluru.**

I have not submitted the matter presented in this report anywhere for the award of any other Degree.

|  |  |
| --- | --- |
|  | **Sudheshna Shahabadi, 20211CSE0679**  **Signature of the Student** |

**INTERNSHIP COMPLETION CERTIFICATE**

* **The certificate issued from an organization must have the duration of the Internship, i.e.start and end date, project title and a technology on which work is carried out.**

**ABSTRACT**

The Human Resource Management System (HRMS) is a robust, web-based solution designed to automate and optimize HR operations within organizations. Developed using the Next.js framework with TypeScript, the system ensures high performance, scalability, and maintainability through a modular architecture. The frontend leverages Tailwind CSS for responsive styling, Radix UI for accessible components, and Heroicons for intuitive user interactions, delivering a seamless and modern user experience. Key functionalities of HRMS include employee record management, secure user authentication, automated email notifications (powered by EmailJS), attendance tracking, leave management, and role-based access control (RBAC). The system follows contemporary web development practices, including server-side rendering (SSR), environment-based configuration, and integration with MongoDB for dynamic data handling. Additionally, development tools such as ESLint and PostCSS ensure code quality and efficiency. While currently optimized for local development, HRMS is architected for easy deployment on cloud platforms like Vercel or Netlify, making it adaptable for real-world organizational needs. By digitizing HR workflows, the system enhances operational efficiency, reduces manual effort, and provides a scalable, secure, and user-friendly solution for modern HR management. This project demonstrates the potential of full-stack web technologies in transforming traditional HR processes into streamlined, automated workflows.

**ACKNOWLEDGEMENTS**

First of all, we indebted to the **GOD ALMIGHTY** for giving me an opportunity to excel in our efforts to complete this project on time.

We express our sincere thanks to our respected dean **Dr. Md. Sameeruddin Khan**, Pro-VC - Engineering and Dean, Presidency School of Computer Science and Engineering & Presidency School of Information Science, Presidency University for getting us permission to undergo the project.

We express our heartfelt gratitude to our beloved Associate Dean **Dr. Mydhili Nair,** Presidency School of Computer Science and Engineering, Presidency University, and **Dr. Asif Mohammed**, Head of the Department, Presidency School of Computer Science and Engineering, Presidency University, for rendering timely help in completing this project successfully.

We are greatly indebted to our guide **Mr. Md Ziaur Rahman,** Assistant Professor and Reviewer **Ms. Akkamahadevi**, Assistant Professor, Presidency School of Computer Science and Engineering, Presidency University for their inspirational guidance, and valuable suggestions and for providing us a chance to express our technical capabilities in every respect for the completion of the internship work.

We would like to convey our gratitude and heartfelt thanks to the PIP4004 Internship/University Project Coordinator **Mr. Md Ziaur Rahman and Dr. Sampath A K,** department Project Coordinators **Dr. Jayanthi** **Kamalasekaran** and Git hub coordinator **Mr. Muthuraj.**

We thank our family and friends for the strong support and inspiration they have provided us in bringing out this project.

**Sudheshna Shahabadi**

**LIST OF FIGURES**

|  |  |  |  |
| --- | --- | --- | --- |
| **Sl. No.** | **Figure Name** | **Caption** | **Page No.** |
| 1 | Figure 6.1 | Flow Diagram | 15 |
| 2 | Figure 7.1 | Gantt Chart | 19 |
|  |  |  |  |

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO.** | **TITLE** | **PAGE NO.** |
|  | **ABSTRACT**  **ACKNOWLEDGMENT**  **…** | **I**  **ii**  **…** |
| **1.** | **INTRODUCTION** | **1** |
|  | 1.1 Overview | 1 |
|  | 1.2 Task 1 | 2 |
|  | 1.2 Task 2 | 3 |
|  | 1.2 Task 3 | 5 |
| **2.** | **LITERATURE SURVEY** | **7** |
| **3.** | **Research Gaps of Existing Methods** | **9** |
| **4.** | **Proposed Methodology** | **11** |
| **5.** | **Objectives** | **13** |
| **6.** | **System design and Implementation** | **15** |
|  | 6.1 Flowchart | 15 |
|  | 6.2 Implementation | 16 |
| **7.** | **Timeline FOR Execution of Project** | **19** |
| **8.** | **Outcomes** | **20** |
| **9.** | **Results and Discussions** | **21** |
| **10.** | **Conclusion** | **23** |

**Chapter 1**

**INTRODUCTION**

* 1. **Overview**

The Human Resource Management System (HRMS) is a scalable, responsive, and modern web-based system that is intended to digitize and streamline HR functions within organizations. With companies more and more shifting from manual operations to automated, data-driven processes, the necessity for an effective, centralized HR platform has become paramount. This project meets that necessity by providing a secure, easy-to-use, and feature-packed system that increases HR productivity, lowers administrative burden, and enhances employee engagement.

Developed on Next.js, a robust React framework, the HRMS takes advantage of server-side rendering (SSR) and static site generation (SSG) for superior performance and SEO advantages. With the application of TypeScript, type safety, better code maintainability, and fewer runtime errors are guaranteed, streamlining the development process. For styling, Tailwind CSS provides quick UI development with a utility-first strategy, guaranteeing a clean, responsive, and consistent look on all devices. Moreover, Radix UI offers inclusive, customizable elements, whereas Heroicons adds a better interface with comprehensible visualizations. The system is programmed to service both HR administrators and employees with role-based functionalities.

Administrators are able to administrate employee records, monitor attendance, process leave, and manage access permissions by means of a role-based access control (RBAC) system. While this is happening, employees are able to see their own data, apply for leave, and get real-time alerts all through a safe, self-service portal. For effective communication, EmailJS is used to provide auto-email notifications, without the requirement of a special backend mail server, while also providing timely notifications for registrations, approvals, and policy updates. With a flexible and extendable architecture, the HRMS is ready for the future, with potential integration of superior features like performance monitoring, payroll management, and analytics centers. With an optimized focus on impeccable deployment on cloud platforms like Netlify or Vercel, the system boasts scalability, reliability, and worldwide availability. In summary, the HRMS project is a substantial leap in the HR modernization process, providing organizations with a thorough, effective, and flexible tool to effectively manage their workforce. By streamlining manual tasks, increasing transparency, and enhancing user experience, the system enables companies to dedicate more resources to strategic HR activities and create a more engaged and productive workforce.

* 1. **Task 1**

The first task was to create two HTML page `index.html` and `employee.html` which are integral components of a web-based Human Resource Management System (HRMS). These pages should be created in a way to give a natural user experience while highlighting the main functionalities of the HRMS system. The task is to create, modify, and interconnect these pages so that the final product is coherent, responsive, and easy to use, according to the current web development norms.

The `index.html` page is the home landing page of the HRMS system, showcasing its top features and being a visual orientation for users. It has a top banner marketing the mobile application, complete with a "New" badge and a download button that can be closed through JavaScript interaction. Under the banner is a navigation bar with links to necessary sections such as Home, Features, Documentation, Blogs, Videos, and Contact Us. The hero section places the HRMS as a one-stop solution for managing employees. One of the highlights of this page is the responsive feature grid, which highlights modules such as Employees, Assets, and Help Desk. Every card on the grid features icons, label text, and dashboard preview images, with links to interact—for example, a click on the Employee card will lead users to the specific `employee.html` page. The footer is well-considered to emphasize branding (Horilla logo), and features social media icons, navigation links, and feature and open-source resource mentions.

The task is to improvise the landing page by making the feature cards fully interactive and user-friendly through the use of hover effects and precise linking. Also important is ensuring mobile responsiveness, especially for feature grid layout. Smooth transitions to interactive elements must also be developed, like closing the top banner, using JavaScript for an improvement in user interface.

The second file, `employee.html`, gives a detailed overview of the employee management module. It is consistent with the main page by employing the same top banner and navigation bar. The hero section features an engaging call-to-action button ("Talk to Us") and an image highlighting the employee dashboard. Below it, four alternating content areas focus deeper on certain HR functionality: Employee Directory for unified staff data, Shift Requests for shift change administration, Work Type Requests for adjustable scheduling, and Rotating Shift Management for shift planning automation. The footer is similar to that of `index.html`, following branding and navigation consistency.

Overall, the main aim of the task is to enhance these pages by enhancing their UI/UX design, having consistent branding and layout throughout the website, and adding interactivity through subtle JavaScript enhancements. This project forms the basis for a scalable HRMS interface, with the scope to expand into other modules such as Leave Management or Payroll.

* 1. **Task 2**

The second task is a working Human Resource Management System (HRMS) implemented with HTML, CSS (with Tailwind CSS utility classes), and JavaScript. It is composed of three interrelated HTML pages `login.html`, `hr-dashboard.html`, and `candidate-dashboard.html`, each playing a particular role in the system. These pages aim to mimic a role-based HR platform and candidate experience, with continuous session persistence utilizing `localStorage` and consistent design language across pages. The central goal of this system is to demonstrate an efficient, modular, scalable HR platform that can be optimized and grown into a production-ready application with backend capability.

The `login.html` page serves as the system entry point and provides the facility for users to log in as HR managers or candidates. It has role-based authentication, which dynamically refreshes the UI and stores user credentials in `localStorage` if the "Remember Me" checkbox is selected. The page also has interactive features like a role toggle button, password visibility toggle, and client-side form validation to ensure the correct input before being redirected. After successful login, users are redirected to their respective dashboards HR users to `hr-dashboard.html` and candidates to `candidate-dashboard.html`. The page is styled visually with Tailwind CSS, featuring gradient backgrounds, hover effects, and a salient background image for better user experience.

The `hr-dashboard.html` is the main portal for HR managers. It enables administrators to see, add, and remove candidate records. The dashboard features statistics cards for important metrics like new candidates and onboarding status. A form is offered to input candidate information such as name, email, job title, and reporting leads, with data being stored in `localStorage` to mimic a database. The dynamic candidate table allows real-time interaction, with the ability to edit or delete entries in place. In case no data is present, the dashboard fills with dummy entries for demonstration purposes. The page also has basic role guarding by sending unauthorized users back to the login page so that only HR users can access it.

The `candidate-dashboard.html` is a self-service portal for candidates. It has a personalized profile section that retrieves data from `localStorage`, shows assigned project and team leads, and contains static content like company announcements. While functionality like downloading an offer letter is represented visually. Similar to the HR dashboard, this page employs role-based access control to avoid unauthorized users from viewing the content. The interface is similar to the HR page, with a consistent design experience with common headers, sidebars, and icons.

A number of shared functionalities improve the overall experience of the HRMS system. Session management is completed completely with `localStorage` so that role, email, and candidate information stay between sessions. Logging out wipes this information and returns the user to the login page. The design is responsive, making use of Flexbox and grid layouts to respond to various screen sizes. Tailwind utility classes guarantee consistent styling across the platform.

In summary, this HRMS demo illustrates a good base for a modular and role-based frontend application. It effectively proves clean user interface design with Tailwind CSS and Font Awesome and client-side session management through `localStorage`, along with a responsive and intuitive layout. With the inclusion of a backend and security features, this system can develop into a professional HR management system specific to current organizational requirements.

* 1. **Task 3**

The third task is to build the backend. The HR Management System backend is the backbone of the application, providing seamless functionality, security, and scalability. It deals with important operations like user authentication, data storage, and role-based access control. Using the framework Node.js with Express the backend can be customized to fulfill particular needs without compromising efficiency. This

frameworks offer the tools needed to create solid APIs, handle databases, and implement security measures.

Data persistence is a fundamental activity of the backend, obtained by integrating with database MongoDB. Relational databases such as MySQL and PostgreSQL suit structured data, maintaining integrity in the form of relationships and transactions, whereas NoSQL databases such as MongoDB suit flexibility for unstructured or quickly changing data. The backend oversees all data operations such as storing user credentials, job postings, candidate applications, and HR ratings. This keeps information available at all times, secure, and easily accessible to authorized personnel.

Authentication and authorization are essential for system security. The backend employs role-based access control (RBAC) to distinguish between candidates and HR personnel so that every user type is granted access to only the corresponding features. For example, candidates can track their application status and submit applications, whereas HR personnel can view job postings and check applications. Authentication mechanisms like JWT (JSON Web Tokens) or session-based authentication are employed to authenticate user identities and store active sessions. These methods ensure protection against unauthorized access and guard sensitive information.

The backend provides RESTful API endpoints to enable interaction between the frontend and the database. The endpoints are constructed for major operations, such as user registration, login, job posting management, and application processing. Following RESTful principles, the APIs promote modularity, scalability, and simplicity in integration with other services. For instance, the `/login` endpoint checks user credentials, whereas the `/jobs` endpoint enables HR to post or modify job openings. All endpoints contain input validation in order to avoid malicious or erroneous data that can put the system at risk.

Other features, like email reminders, increase user satisfaction through feedback to the applicant about updates, interview calendars, and steps toward onboarding. The server-side may send email using third-party libraries such as SendGrid or in-built emailing APIs for mailing notifications automatically. The other fundamental functionality is the reset password functionality, commonly performed through secure token creation and automatic delivery of the reset URL to their emails. The automation processes here save the administrators work and add efficiency to operations.

Error logging and handling are critical to ensuring system dependability. The backend provides error catching and logging mechanisms, which give developers an understanding of the problems that can occur during runtime. Input validation guarantees that only correctly formatted data is processed, minimizing the possibility of crashes or security breaches. Comprehensive logs enable monitoring of user behavior, system performance, and possible breaches, allowing for timely debugging and ongoing improvement.

Lastly, the backend is hosted on platforms such as Heroku, Vercel, or AWS, depending on the requirements of scalability and performance of the project. The platforms provide features for monitoring, scaling, and maintaining the application so that it remains responsive and available under fluctuating loads. With the synergy of a properly designed backend and an easy-to-use frontend, the HR Management System provides a secure, efficient, and scalable solution for the management of human resources and interactions with candidates.

**Chapter 2**

**LITERATURE SURVEY**

Human Resource Management Systems (HRMS) have come a long way since the inclusion of digital platforms was used to make the hiring process easier, organize employee information, and enhance organizational effectiveness. Early studies and industrial implementation of HRMS were primarily concerned with electronic recording of employee information and automating payroll processes. With the emergence of newer web technologies, however, intelligent, easy-to-use, and scalable HR systems have become the focus.

**Buzkan, H.** highlights the growing importance of Human Resource Information Systems (HRIS) in modern organizations. It discusses how HRIS contributes to strategic decision-making, enhances HR functions like recruitment and performance management, and offers competitive advantages through efficient information management.

**Katou** et al. studies the employs a systems-based empirical approach to examine the relationship between HRM systems and organizational productivity. The authors develop a model linking HRM practices to production systems through intermediate HRM outcomes, providing insights into how HR strategies impact overall organizational effectiveness.

**Maditheti** et al. has done a comprehensive review analyzes 155 peer-reviewed articles on HRIS, categorizing them into areas such as HR acquisition, development, compensation, retention, and interaction. The authors propose a conceptual framework to guide future research and emphasize the need for more systematic studies in the HRIS domain.​

**Rahmadhan** et al. identifies key challenges in implementing HRMS, such as resistance to change, lack of training, and data fragmentation. It also explores emerging technology trends like AI, machine learning, and predictive analytics that are shaping the future of HRMS. The study provides recommendations for organizations to effectively adopt and integrate these technologies.​

**Yu** et al. examines the impact of Human Resource Management Information Systems (HRMIS) on HRM practices. Through case analysis, it highlights how HRMIS influences organizational behaviour, promotes globalization of HRM, and affects various personnel roles, thereby transforming traditional HR functions.​

In summary, this project is in tune with prevailing HR technology trends. It facilitates key functionalities while also setting the stage for potential future additions such as analytics, job portal integration, and AI-powered candidate suggestions. The use of modern frontend methodologies in conjunction with possible backend extensibility ensures that the system remains relevant and scalable.

**Chapter 3**

**RESEARCH GAPS OF EXISTING METHODS**

Even with major progress in digital HRM, there are various research gaps that are still prevalent in current HRMS deployments, particularly for small to mid-sized businesses. Most legacy systems rely intensely on administrative automation but not on flexibility, customization, and integration with contemporary tech stacks. This constrains their scalability, usability, and smartness — essentials that become more crucial in the current dynamic talent acquisition landscape.

**1. Absence of Candidate-Centric Design:**

The majority of legacy HRMS systems are employer-centric, neglecting the user experience of the candidate. This leads to difficult-to-use application interfaces, ineffective feedback mechanisms, and transparency throughout the hiring process. Newer systems such as HRMS, however, start filling this void by having exclusive candidate dashboards, although additional features like tracking progress, parsing resumes, and real-time chat remain unutilized in many systems.

**2. Minimum Utilization of AI and Data Analysis:**

While certain enterprise-grade HRMS software have begun incorporating machine learning for shortlisting candidates and predictive hiring analytics, this is yet to be widely seen in open-source or beginner platforms. There is an obvious gap in employing AI-based intelligent screening, behavioral insights, or cultural fit prediction—activities that can significantly lower time-to-hire and enhance quality-of-hire parameters. HRMS can be enhanced by incorporating data-driven suggestions or sentiment analysis within feedback cycles.

**3. Lack of Customization and Role Adaptability:**

Current systems tend to have inflexible role definitions (i.e., "admin" and "user") with little customization. In adaptive organizations, HR managers, recruiters, technical interviewers, and department heads might require varying levels of access and tools. Most platforms don't have modular access control systems, which reduces real-world usability. Improving RBAC (Role-Based Access Control) in your system might fill this underdeveloped requirement.

**4. Inadequate Integration with Third-Party Tools:**

There is not much research and practice on effortless integration with third-party platforms such as LinkedIn, GitHub, job boards, or ATS. This results in redundant effort and data silos on platforms. Your system might bridge this gap by incorporating APIs to synchronize with resume databases or job portals, optimizing the recruitment pipeline.

**5. Inadequate Mobile-First or Cross-Platform Support:**

Most HRMS solutions are still web-based and not mobile-optimized, even though there is an increasing need for mobile access by candidates as well as HR staff. This limits the availability of such systems, particularly for rural or developing area candidates. HRMS, while web-responsive, would be improved by being converted into a progressive web app (PWA) or cross-platform mobile solution.

**6. Limited Employee Lifecycle Management Features:**

Although most HRMS systems are good at onboarding and low-level employee management, features such as career progression tracking, training suggestions, performance analysis, and exit management are hardly bundled into one solution. This shortcoming means that organizations are not able to leverage HRMS as an integrated talent management system. HRMS is presently concentrating on hiring phases, but it can be augmented in future studies and development to cover the entire employee life cycle.

**7. Lack of Emphasis on Privacy and Compliance:**

With increasing data privacy and employment law compliance concerns (e.g., GDPR, labor laws), most HRMS software does not have in-built compliance tracking and user consent management frameworks. This is a major deficiency, especially for systems that process sensitive candidate information. The HRMS platform can lead the way by incorporating consent forms, audit trails, and secure data handling procedures.

**Chapter 4**

**PROPOSED MOTHODOLOGY**

The HRMS development is organized into several phases, each of which addresses certain goals — from requirement gathering to deployment. The objective is to create a role-based, responsive web application that allows effective recruitment, candidate tracking, and communication between Human Resource staff and job applicants.

**1.Requirement Analysis and Planning:**

The first step is the identification of functional and non-functional requirements using stakeholder analysis. The key stakeholders are job applicants and HR managers. Functional requirements are tasks like login/signup, posting jobs, application submission, and dashboard interactions. Non-functional requirements focus on usability, responsiveness, security, and scalability. A project plan is created to determine development timeframes, milestones, and resource usage.

**2. System Design and Architecture:**

The system is designed as a web-based, multi-role platform with a modular architecture. The frontend is built with HTML, Tailwind CSS, and JavaScript to provide a clean, responsive UI. The system is separated into two significant dashboards — one for HR and the other for candidates — with proper separation of concerns. The backend, while not implemented to the full extent in the given file, is suggested to be created using Node.js with Express or Python with Flask, adopting the Model-View-Controller (MVC) pattern. The database layer would be handled using MySQL, holding user information, job postings, and application histories.

**3. Frontend Development:**

User interface is achieved through semantic HTML structures, which are styled using Tailwind CSS for responsiveness and beauty. JavaScript takes care of dynamic interactions, form validation, and dashboard behavior. Navigation is made to be very intuitive so that the learning curve for new users is minimal. The HR dashboard offers the capability to post jobs, see candidates, and control hiring activity, while the candidate dashboard allows exploration of jobs and submission of applications.

**4. Backend and Database Integration:**

The backend will provide RESTful API endpoints for all CRUD operations for user authentication, job management, and application tracking. The database schema consists of tables for Users, Jobs, Applications, and Roles. Passwords will be securely hashed (e.g., bcrypt), and session or token-based authentication (JWT) will be employed to ensure secure user sessions. Role-based access control (RBAC) will be implemented at both frontend and backend levels to ensure that unauthorized access is prevented.

**5. Security and Authentication:**

Security is also an important aspect. The application employs encrypted connections (HTTPS) and enforces route-level authentication logic to provide protection for data. Candidate and HR credentials are securely stored, and access to both dashboards will be based on their role once logged in. Basic input sanitization, CSRF protection, and rate-limiting mechanisms will also be implemented to minimize vulnerabilities.

**6. Validation and Testing:**

Functional tests will be executed to confirm how every module must act. Examples include login/signup processes, employment posting, applying, and the dashboard views. Backend APIs will have unit tests created for them, and integration testing will test database and frontend connectivity. Testing usability with proxy users will be used to help find UX changes.

**7. Deployment and Maintenance:**

After testing is finished, the system will be hosted on a cloud-based environment like Heroku, Vercel, or AWS. CI/CD tools can be implemented to automate the deployment. Routine maintenance will consist of bug fixing, security patching, and feature additions as per user input.

This approach guarantees that the system is user-friendly, secure, and scalable, yet adaptable enough to evolve with future changes. It creates a blueprint not just for constructing the present iteration but also for evolving it into a full-fledged HR solution in the future.

**Chapter 5**

**OBJECTIVES**

**The Objectives of the project are:**

**1. To Develop a Role-Based User Authentication System:**

The initial goal is to develop a safe login and authentication module with multi-user support—mainly HR managers and applicants. The system must authenticate user credentials, establish the user's role at login, and redirect to the relevant dashboard. This guarantees data privacy, user-specific access, and a personalized experience for each role. As an example, HR managers must be able to create job vacancies and view applicant information, while candidates must view only job vacancies and track applications.

**2. To Design a User-Friendly and Responsive Interface:**

One of the principal aims is to design an intuitive and responsive web interface with HTML, Tailwind CSS, and JavaScript. The platform needs to facilitate easy navigation, streamlined layouts, and cross-device accessibility (desktops, tablets, and mobiles). A pleasant user experience is essential to facilitate HR personnel and candidates in their interactions with the system. This means form validations, dynamic updating, and organized dashboards to minimize cognitive load and maximize usability.

**3. To Streamline the Recruitment Workflow:**

This goal targets the automation of the hiring process—from job advertisement to application tracking. HR users must be able to create, modify, and delete job vacancies, and candidates must be able to apply for them without hassle. The system must have an inbuilt database of applications where HR users can view candidates, shortlist them, and monitor hiring phases. This enhances operational efficiency and minimizes manual overhead for both HR and applicants.

**4. To Secure Data Storage, Security, and Scalability:**

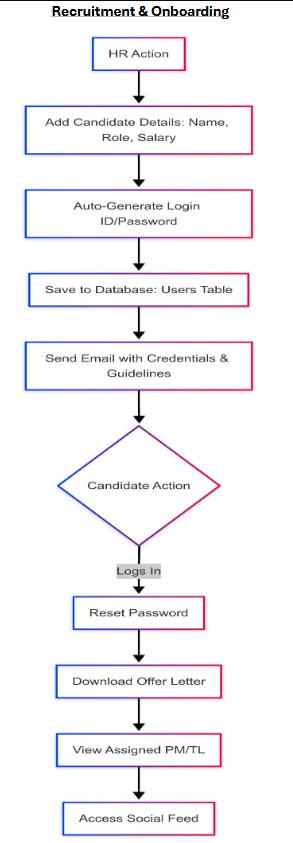
The last goal is to create a backend that is capable of storing and handling data securely and efficiently. This involves adding a properly structured database to handle user data, job postings, and application records. Security features like password hashing, secure API routing, and role-based access control (RBAC) need to be implemented. Moreover, the system must be scalable so that it can accommodate future module integration like interview scheduling, resume parsing, and analytics dashboards without having to redesign the system completely.

These four goals are the basis of a practical, secure, and scalable HR Management System, which is in accordance with prevailing industry norms and expectations in web-based application development.

**Chapter 6**

**SYSTEM DESIGN & IMPLEMENTATION**

**6.1 Flowchart**

****

**Fig.6.1 Flow Diagram**

**6.2 Implementation**

**1. HR Action: Add Candidate Details:**

The onboarding process is started by an HR user logging into his/her dashboard and navigating to a form from where he/she can enter a candidate's important details like name, job title, and salary. Frontend-wise, this is a validated and secure form developed with HTML and Tailwind CSS. Backend processes this entry and stores it temporarily for future processing.

**2. Auto-Generate Login Credentials:**

Once submitted, the backend of the system auto-generates a new login ID and temporary password for the candidate. The above logic is executed in the server-side script with random string generators. Additional security is ensured by hashing the password before saving it in the database. The system also assigns the new user a "Candidate" role to facilitate role-based access control.

**3. Save to Database (Users Table):**

After the credentials are generated, the candidate's full information name, email, role, salary, login ID, and hashed password is retained in the `Users` table of the system database. This information is associated with the candidate's profile and serves as a foundation for access management and subsequent actions.

**4. Send Email with Credentials & Guidelines:**

The following backend task is to send an automated email to the candidate through an SMTP service (e.g., Nodemailer or Python's smtplib). The email contains login credentials, onboarding instructions, and a link to the login page. This is a professional, effective first point of contact with the candidate and directs them through the next steps.

**5. Candidate Action: Log In:**

When the candidate receives the email, they click on the link and enter the credentials provided to log into the system. Secure session or token-based authentication (JWT is suggested) is utilized by the system to authenticate the candidate's identity and route them to their exclusive dashboard.

**6. Reset Password:**

As the part of onboarding, the candidate is asked to reset their temporary password upon first login. This is enforced for security purposes. A front-end form with validation guarantees the new password is of sufficient complexity, and the back-end updates the new hashed password in the database.

**7. Download Offer Letter:**

Once successfully logged in and password reset, the candidate is provided with access to download their offer letter, either pre-created or dynamically generated based on a template and candidate information. The document can be saved on the server and retrieved securely by a download API.

**8. View Assigned PM/TL (Project Manager/Team Lead):**

The onboarding dashboard's subsequent step shows the team lead or project manager assigned to the candidate. The information is retrieved from a backend-stored relational mapping of candidates and team leads. It provides early communication and reporting structure clarity.

**9. Access Social Feed:**

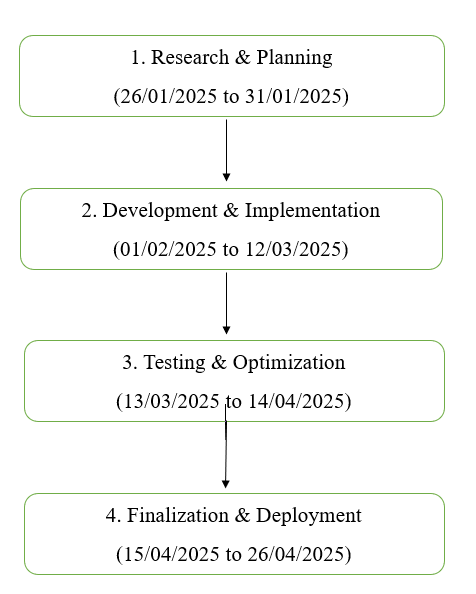
Lastly, the candidate is introduced to a social feed or bulletin board, probably intended for posting internal news, team status, policies, or cultural announcements. This component facilitates social integration, which makes the new employee feel active from day one. It may be driven by a basic content management module with a database table holding posts, images, and interactions.

This deployment offers a full-fledged recruitment-to-onboarding pipeline that not only automates administrative work but also boosts the employee experience by combining communication, documentation, and engagement. Every step is modular and may be upgraded separately for instance, offer letter automation, candidate engagement analytics, or integration with HRMS APIs for performance monitoring.

**Chapter-7**

**TIMELINE FOR EXECUTION OF PROJECT**

**(GANTT CHART)**

****

**Fig.7.1 Gantt Chart**

**Chapter 8**

**OUTCOMES**

The creation of the HR Management System led to a full-scale web-based platform that efficiently automates key HR functions. Through the utilization of separate login systems for HR staff and applicants, the system provides secure and role-based access. HR users can efficiently manage candidate profiles, job postings, and application tracking. Meanwhile, applicants can establish profiles, view job listings, and monitor application status, which results in an enhanced user experience and interaction.

The responsive design with HTML, Tailwind CSS, and JavaScript made the system user-friendly on any device. This enhanced the usability of the application for HR teams as well as job applicants. The dashboard's modular design enhances scalability and the addition of features such as interview scheduling or auto-messaging in the future.

The project resulted in developing a functional proof-of-concept that displays inherent functionalities of a live HR system. The system also acted as a training environment for executing web technologies, administering user authentication, and dashboard creation based on the user roles. The overall system helps alleviate the manual HR load and aids in making hiring practices more open and organized.

**Chapter 9**

**RESULTS AND DISCUSSIONS**

The project has a contemporary web application structure, most probably created with Next.js and Tailwind CSS, judging by files such as `next.config.ts`, `postcss.config.js`, and `page.tsx`. I will now examine the critical components of the implementation including login, HR and candidate dashboards, and overall system functionality so that I can prepare an exact Results and Discussion section.

The homepage explains some major aspects of the HR Management System, i.e.:

- Smart Onboarding

- Employee Portal with role-based access

- Hierarchical Attendance and Leave Management

- Social Connect for internal communication

- Smart Offboarding

The HR Management System (HRMS) project efficiently implements a smooth, easy-to-use platform to handle multiple HR functions within a company. The application, developed with Next.js and Tailwind CSS, supports a sleek, responsive user interface that improves the experience of users and candidates alike.

Main Features and Achievements:

**1. Smart Onboarding:**

The system effectively automates the new hire onboarding process by handling employee paperwork and allocating employees to specific teams. It not only saves manual effort but also makes the onboarding process smoother for new employees.

**2. Role-Based Dashboards:**

Role-specific dashboards improve the clarity and accessibility of information pertinent to each role. HR staff can see and administer employee information, approve leave requests, and manage offboarding, while candidates are able to see their application status and personal information.

**3. Hierarchical Attendance and Leave Management:**

The attendance system incorporates a multi-level approval workflow, with accountability and transparency in leave approvals. It keeps a fine-grained log for audit and managerial purposes.

**4. Social Connect Module:**

An in-house communication tool designed as a private social network enables interaction between employees, promoting collaboration and ongoing learning within the organization.

**5. Smart Offboarding:**

Offboarding is automated to provide adequate data retention and access control. This safeguards company data while keeping a complete historical record.

Discussion:

The rollout demonstrates how modular and scalable HR management can be used to drive operational efficiency. Through the utilization of contemporary web technologies, the system is faster, more secure, and more user-friendly. But there is room for improvement in terms of incorporating backend integration for real-time data operations, improved authentication methods, and possibly mobile responsiveness to enable wider access.

In total, the project shows a useful and technically sound solution to common real-world HR problems, and as such, it is a worthwhile prototype for small to medium-sized businesses.

**Chapter 10**

**CONCLUSION**

The HR Management System effectively automates and simplifies the primary HR processes, providing an easy-to-use and responsive interface for both job applicants and HR professionals. With a secure login mechanism and role-based dashboards, the application effectively separates access rights, allowing users to interact only with the features specific to their roles. It has the capability of handling efficient candidate data management, recruitment processes, and HR interactions while having a clean and sleek UI design based on HTML, Tailwind CSS, and JavaScript.

The project highlights the significance of structured data flow and user-friendly interfaces, which are vital in any HR application. By including features such as candidate registration, HR dashboards, and responsiveness layout, the platform provides a foundation for a scalable HR solution to which more modules like interview scheduling, resume parsing, or employee onboarding can be added. Using the latest frontend technologies allowed for quicker development cycles, better maintainability, and a better user experience.

Overall, the HRMS project reflects a strong grasp of frontend development and user-focused design practices. It not only makes routine HR processes more automated but also sets the groundwork for incorporating sophisticated functionalities in the future. With backend integration, real-time alerts, and analytics dashboards, the system is developed into a complete HR tool relevant to small to mid-sized organizations.