An Internship Report

On

STUDENT ATTENDANCE TRACKING SYSTEM

Submitted in partial fulfillment of the requirement for the award of the degree of

BACHELOR OF TECHNOLOGY

In

ELECTRONICS AND COMMUNICATION ENGINEERING

Submitted by

T.N.S.Gayathri	**(21A91A04P3)**
P.Arjuna Rao	**(21A91A04O4)**
K.Divya Lakshmi	**(21A91A04K8)**
M.Chandu	**(21A91A04M5)**
P.Sai Yasaswi	**(21A91A04N4)**
O.Devi Priyanka	**(21A91A04N3)**

DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING

ADITYA ENGINEERING COLLEGE(A)

Approved by AICTE – Permanently Affiliated to JNTU Kakinada – Accredited by NBA, NAAC with A++ Grade, Recognized by UGC under sections 2(f) and 12(B) of UGC Act 1956 Aditya Nagar, ADB Road - Surampalem, E.G.Dist., A.P-533437 (2023-2024)

ADITYA ENGINEERING COLLEGE

(Affiliated to JNTU Kakinada and Approved by AICTE, New Delhi)

DEPARTMENT OF
ELECTRONICS & COMMUNICATION ENGINEERING

Certificate

This is to certify that the internship report entitled “STUDENT ATTENDANCE TRACKING SYSTEM” being submitted by **T.N.S.Gayathri(21A91A04P3),P.ArjunaRao(21A91A04O4), K.DivyaLakshmi(21A91A04K8),M.Chandu(21A91A04M5),P.SaiYasaswi(21A91A04N4),O.Devi Priyanaka(21A91A04N3)** for the partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING to the Jawaharlal Nehru Technological University, Kakinada is a record of bonfide work carried out by them under the guidance and supervision.

To the best of our knowledge, the results embodied in this project report have not been submitted to any other University or Institute for the award of degree.

HEAD OF THE DEPARTMENT

DR.Sanjeev Kumar

Professor

Acknowledgement

The satisfaction that accompanies successful completion of any task would be in-complete without the mention of the people who made it possible and whose constant guidance and encouragement crown all the efforts with success. The acknowledgement transcends the reality of formality.

We are thankful to our beloved MR.GANAPATHI DURGA SAI PRASAD Web Developer who has spared her valuable time and append novel ideas to guide us in lime light. We are indebted to his without whom we not have culminated to the pinnacle of this project.

We also wish to convey our sincere thanks to MR.BABJI NEELAM, Founder of Technical Hub who provided the place and requirements which was necessary of the project.

We also wish to convey our sincere thanks to DR. SANJEEV KUMAR, Head Of the Department who provided vital information which was necessary of the project.

We are thankful to DR. D.SANJAY, Principal. Aditya Engineering College for providing appropriate environmental required for this project.

We are also thankful to all staff members, parents and friends who are directly and indirectly helped us in the completion of the project with flying colors.

ABSTRACT

Training and attendance tracking helps keep track of who attends classes or workshops and their progress. It's like a digital attendance sheet that also records what people learn and how they're doing. This helps organizers know who's participating and how effective the training is.

Key Components and Features:

- **Real-Time Attendance Monitoring**: Enables teachers and administrators to track attendance in real-time, , ensuring immediate updates and oversight.
- **User-Friendly Interface**: Provides an intuitive interface for students, teachers, and administrators to easily mark, view, and manage attendance records.
- **Data Security and Privacy**: Ensures that all attendance data is securely stored and complies with privacy regulations, protecting student information.
- **Customizable Settings**: Allows institutions to customize attendance policies, thresholds for alerts, and reporting formats to meet their specific needs.
- **Integration with Existing Systems**: Seamlessly integrates with existing school management systems, learning management systems (LMS), and other educational software for streamlined operations.
- **Multi-User Access**: Supports role-based access, ensuring that teachers, administrators, and students have appropriate access levels and functionality.
- **Offline Functionality**: Offers offline attendance marking capabilities, with automatic syncing once internet connectivity is restored.

INDEX

CHAPTER 1

Technology Stack

The project utilizes a comprehensive technology stack, incorporating front-end frameworks like React or Angular, back-end technologies such as NodeJs or Django, and robust databases like MySQL or MongoDB. Additionally, it leverages cloud services from providers like AWS for scalable and secure data storage and processing.

This integration ensures a seamless, efficient, and reliable student attendance tracking system.The use of RESTful APIs facilitates smooth communication between components, and CI/CD pipelines ensure continuous integration and deployment for rapid development and updates.

Frontend Technologies:

1. The frontend of the Student Attendance Tracking System is developed using ReactJs, a powerful JavaScript library that enables the creation of dynamic and responsive user interfacesthrough reusable components and a virtual DOM.
2. This ensures a smooth, efficient, and interactive user experience across devices.

Back-end Technologies:

1. The backend of the Student Attendance Tracking System uses NodeJs with ExpressJs, providing a robust environment for handling asynchronous operations and building scalable APIs.
2. This setup ensures efficient data processing and real-time interaction capabilities.

Data Base:

1. The Student Attendance Tracking System utilizes MongoDB, a flexible NoSQL database that stores data in JSON-like documents, allowing for dynamic and scalable data handling.
2. This ensures efficient storage and retrieval of diverse and evolving data structures.

Devops:

1. The Student Attendance Tracking System is hosted on AWS, leveraging its scalable cloud infrastructureto ensure high availability and performance.
2. AWS services provide secure and reliable storage, compute power, and networkcapabilities for the application.
3. Docker containers enable containerized deployments for microservices architecture,promoting scalability and portability.

The chosen technologies contribute to:

Front-End Technologies (ReactJs):

1.Component-Based Architecture: They allow for reusable components, making the development process more efficient and the codebase easier to maintain.

Back-End Technologies (NodeJs):

1. **High Performance**: NodeJs offers non-blocking, event-driven architecture, which is ideal for real-time applications, while Django provides a high-level framework that encourages rapid development.

Databases (MongoDB):

1. **Flexibility**: MongoDB's schema-less design allows for the storage of unstructured data, providing flexibility in data modeling and quick iteration.

Cloud Services (AWS):

1. **Scalability**: Cloud services from AWS provide auto-scaling features, ensuring the system can handle varying loads efficiently.

CHAPTER 2

Frontend and Backend Installation

To install front-end technologies like React use package managers such as npm or yarn to run commands like `npx create-react-app my-app`. For back-end technologies like NodeJs or Django, install the necessary packages using npm with `npm install express` for NodeJs .

- npx create-react-app app_name
- npm config set legacy-peer-deps
- npm i --legacy-peer-deps
- npm cache clean --force
- npm i styled-components
- npm i react-router-dom
- npm i axios
- npm i react-bootstrap bootstrap
- npm init
- npm i express
- npm i mongoose
- npm i cors
- npm i body-parser

React Packages:

1. **react-router-dom**: This module provides routing capabilities for React applications, including components like BrowserRouter, Route, Switch, and Link to enable navigationbetween different views or pages within the application.
2. **axios**: Axios is a popular HTTP client for making asynchronous HTTP requests in Reactapplications. It simplifies the process of sending and handling HTTP requests to backendservers or APIs.
3. **redux** and **react-redux**: Redux is a predictable state container for JavaScript apps, commonly used with React for managing application state. react-redux provides

bindings to use Reduxwith React components, enabling efficient data flow and state management across the application.

4. **styled-components**: This module allows you to write CSS-in-JS in your React components. Itenables the creation of styled components with scoped styles that are easy to maintain and reuses.
5. **react-icons**: This library offers a collection of popular icons as React components, allowingeasy integration of icons from various icon sets into your application.

React Modules:

1. **Components**: Modular building blocks that encapsulate UI elements and logic, promotingreusability and maintainability.
2. **JSX**: Syntax extension for JavaScript that allows HTML-like code within React components,facilitating the creation of UI elements.
3. **Props**: Short for properties, props are used to pass data from one component to another,enabling dynamic and flexible component behavior.
4. **State**: State allows components to manage and store internal data, enabling them to updateand render based on changing conditions or user interactions.
5. **Hooks**: Introduced in React 16.8, hooks like useStateand useEffectprovide a way to use state and lifecycle features in functional components, improving code organization and reusability.

HTTP Methods:

HTTP methods provide a standardized way for clients (such as web browsers or mobile apps) tointeract with backend servers and perform CRUD (Create, Read, Update, Delete) operations on resources. Proper use and adherence to HTTP methods help maintain clarity, consistency, and scalability in API design and development.

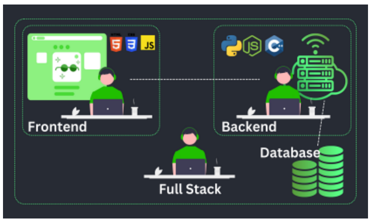
- **GET**: Retrieves data from a specified resource.
- **POST**: Submits data to be processed to a specified resource.
- **PUT**: Updates a specified resource with new data.
- **DELETE**: Deletes a specified resource.

Backend Packages:

- **Express**: This is the core web application framework for Node.js. It simplifies the process of handling HTTP requests, defining routes, and managing middleware. Express is crucial for buildingrobust APIs and web applications.
- **Mongoose**: Mongoose is an ODM (Object-Document Mapper) for MongoDB and Node.js. It provides a schema-based solution to model application data, perform CRUD operations, and interactwith MongoDB databases using JavaScript objects.
- **Cors**: CORS (Cross-Origin Resource Sharing) middleware allows control over how resources fromdifferent origins (domains) can interact with your server. It's essential for handling requests from frontend applications that are hosted on different domains than your backend.
- **body-parser**: While included in Express.js as express.json()in newer versions, body-parserwas traditionally used to parse incoming request bodies in middleware. It's crucial for accessing POST, PUT, and PATCH request parameters or JSON payloads in your routes.

Schemas (MongoDB):

1. **Mongoose Schema**:In a Node.js application using MongoDB with Mongoose, schemas define the structure of documents within a collection. They specify the fields, their types,validation rules, and default values.



The installation process for both front-end and back-end technologies is straightforward and well-documented, ensuring developers can quickly set up their development environments. Using package managers like npm, yarn, and pip simplifies dependency management, allowing for efficient project initialization and configuration.

CHAPTER 3

Features

Login and Signup

- **User Authentication:** The system includes a robust authentication mechanism, providing secure login and signup functionalities for both employees and students. Each user must register through a simple signup process, which captures essential details like name, email, contact number, and role (student or employee). Post-registration, users can log in using their credentials, which ensures that only authorized personnel can access the system.
- **Role-Based Access Control:** The login module is designed with role-based access control. This ensures that students, teachers, and administrators have access to different sets of features based on their roles. For instance, students can only view their attendance records, whereas teachers can mark attendance and generate reports. Administrators have access to all the functionalities, including user management and system settings.
- **Security Measures:** To protect user data, the system employs encryption techniques for passwords and sensitive information. Multi-factor authentication can be implemented as an additional security layer, requiring users to verify their identity through an email or SMS code during login.

Employee and Student Management

- **User Profiles:** Once logged in, employees and students have access to their profiles. Employees can update their contact details, manage class schedules, and view student attendance. Students can view their attendance history, apply for leave, and receive notifications about their attendance status.
- **Administrative Control:** Administrators can add, remove, and manage user accounts for both employees and students. This includes assigning roles, resetting passwords, and updating personal information. The system also allows for batch processing of user accounts, making it easy to handle large numbers of users.
- **Attendance History:** The system maintains a comprehensive record of each student's attendance. Students can view their daily, weekly, and monthly attendance summaries. This feature helps students keep track of their attendance and identify any discrepancies.

Branch-Wise Filtering

- **Organizational Structure:** The system supports branch-wise filtering, allowing institutions with multiple branches to manage attendance data separately for each branch. This is particularly useful for universities and colleges with different campuses or departments.
- **Branch-Specific Access:** Administrators can assign users to specific branches, ensuring that teachers and students have access only to the relevant branch data. This enhances data security and ensures that branch-specific operations are streamlined.
- **Data Segmentation:** The branch-wise filtering feature segments attendance data based on branches, making it easier to generate branch-specific reports. This helps in analyzing attendance patterns across different branches and implementing targeted interventions to improve attendance rates.

Daily Attendance Tracking

- **Real-Time Attendance:** Teachers can mark student attendance in real-time using the system. This can be done through various methods, including manual entry. Real-time tracking ensures immediate updates to the attendance records.

The Student Attendance Tracking System is designed to simplify and enhance the management of student attendance through its comprehensive features. By incorporating secure login and signup functionalities, role-based access control, and branch-wise data segmentation, the system ensures efficient and organized handling of attendance records. Real-time tracking, automated alerts, and detailed reporting further streamline the process, providing valuable insights and facilitating better administrative control. This system not only improves accuracy and efficiency but also promotes a proactive approach to managing student attendance, ultimately contributing to better educational outcomes.

CHAPTER 4

Deployment on Cloud

Deploying the Student Attendance Tracking System on the cloud ensures high availability, scalability, and secure data management. Here's a detailed overview of the cloud deployment process:

Cloud Service Provider Selection

AWS, Azure, or Google Cloud: Choose a cloud service provider based on your specific requirements. AWS, Azure, and Google Cloud offer a range of services suitable for deploying web applications, including compute, storage, and networking capabilities. Each provider has its own set of advantages, pricing models, and support options.

Setup and Configuration

- **Virtual Machines and Containers:** Use virtual machines (VMs) or containers (Docker) to host your application. Containers are particularly useful for creating consistent environments across different stages of deployment and simplifying the scaling process.
- **Compute Services**: Utilize compute services like Amazon EC2 (AWS), Azure Virtual Machines, or Google Compute Engine to run the backend of the application. For containerized applications, consider using services like AWS ECS/EKS, Azure Kubernetes Service (AKS), or Google Kubernetes Engine (GKE).
- **Storage and Databases:** Choose cloud-based storage and database services for managing your data. Options include Amazon RDS or DynamoDB, Azure SQL Database or Cosmos DB, and Google Cloud SQL or Firestore. These managed services offer high availability, automated backups, and easy scaling.

Deployment Steps

- **Code Repository Integration:** Use a code repository like GitHub, GitLab, or Bitbucket to manage your application code. Integrate the repository with your cloud service to enable continuous integration and continuous deployment (CI/CD).

- **CI/CD Pipeline Setup:** Set up a CI/CD pipeline using services like AWS CodePipeline, Azure DevOps, or Google Cloud Build. This pipeline automates the process of building, testing, and deploying your application, ensuring quick and reliable updates.
- **Environment Configuration:** Configure different environments (development, staging, and production) in your cloud service. This includes setting up environment variables, configuring security groups, and managing permissions.
- **Deployment Automation:** Use infrastructure as code (IaC) tools like AWS CloudFormation, Azure Resource Manager (ARM) templates, or Terraform to automate the deployment and management of your cloud infrastructure. This ensures consistency and repeatability in your deployment process.

Deploying the Student Attendance Tracking System on the cloud offers numerous benefits, including scalability, high availability, and enhanced security. By leveraging cloud services, you can efficiently manage and scale your application to meet the needs of your users while ensuring robust data protection and compliance. This cloud-based approach not only streamlines deployment and maintenance but also provides a flexible and reliable infrastructure to support the long-term success of your system.

CHAPTER 5

CONCLUSION

The Student Attendance Tracking System presents a comprehensive solution for modernizing and streamlining the attendance management process in educational institutions. By incorporating advanced technologies such as biometric and RFID systems for real-time attendance tracking, user-friendly interfaces, and automated notifications, the system significantly enhances efficiency, accuracy, and transparency. These features ensure that students, teachers, and administrators can effectively manage and monitor attendance, leading to improved student engagement and institutional oversight.

Deploying this system on the cloud further amplifies its benefits by offering unparalleled scalability, security, and reliability. Cloud platforms such as AWS, Azure, and Google Cloud provide the necessary infrastructure to handle the dynamic needs of educational institutions, allowing for seamless scaling and robust data management. The integration of front-end technologies like React and Angular with back-end frameworks such as Node.js and Django, along with secure cloud storage solutions, ensures that the system remains performant and secure, safeguarding sensitive data and adhering to compliance standards.

Overall, the Student Attendance Tracking System represents a significant step forward in leveraging technology to enhance educational administration. Its comprehensive features, combined with the advantages of cloud deployment, provide a powerful tool for improving attendance management and fostering a more efficient, responsive, and data-driven educational environment. This system not only addresses current challenges but also lays a foundation for future advancements in educational technology, ultimately contributing to better educational outcomes and experiences for all stakeholders.