Debre Berhan University

College of Computing

Department of Software Engineering

Internship Report

Project Title: Complaint Management System .

Hosting Company: AddisWay Software Technology P.L. .

By: Abdulaziz Musa Hussen (DBUR/0597/13)

Mentor: Hiwot K.

Company Supervisor: Muhammed

November, 2024

Table of Contents

Acknowledgment

I would like to extend my heartfelt gratitude to AddisWay Technology Solutions P.L.C. for the invaluable opportunity to intern and grow within a professional environment. My deepest appreciation goes to my supervisor Mr. Mohammed and my mentor Ms. Hiwot who provided patient guidance and valuable insights throughout the course of the Complaint Management System project. Their expertise and mentorship played a key role in enhancing my technical skills and deepening my understanding of software development.

I am also grateful to my colleagues, whose collaboration, feedback, and encouragement contributed significantly to my learning experience. Their teamwork and support fostered a positive work environment, allowing me to confidently take on challenges and develop both my technical and interpersonal skills.

This internship experience has been pivotal to my personal and professional growth, and I am sincerely thankful to everyone at AddisWay Technology Solutions and to at my University who played a role in making this journey meaningful and fulfilling.

Part One: Internship Experience

Executive Summary

This report provides a comprehensive overview of my internship experience at AddisWay Technology Solutions P.L.C., a leading technology solutions provider based in Addis Ababa, Ethiopia. The primary objective of my internship was to gain practical experience in software engineering, specifically working on web and mobile application development using the MERN (MongoDB, Express.js, React, and Node.js) stack. My tasks included working on software design, developing and testing components, and assisting with project implementation. The internship was intended to enhance my technical skills and apply theoretical knowledge to real-world projects, aligning with both my academic studies and career goals in software development.

The methodology employed throughout the internship involved Agile development practices, which included sprint-based task management, regular feedback sessions, and collaborative problem-solving. These practices enabled a structured yet flexible approach to development, allowing for continuous learning and iteration.

The expected output of this internship was to deliver well-tested and functional software components for client projects, while simultaneously achieving personal growth in technical skills and project management capabilities. Beneficiaries of the project included both the company and its clients, as my work contributed to improving the company's project efficiency and client satisfaction. The internship also benefited my professional development, providing valuable experience in a dynamic, real-world setting.

1. Introduction

AddisWay Technology Solutions P.L.C. is a prominent technology solutions provider based in Addis Ababa, Ethiopia, focused on delivering innovative software and IT services to businesses and organizations. The company's mission is to empower clients with digital tools and solutions that enhance efficiency and competitiveness. AddisWay's core activities include custom software development, IT consulting, and digital transformation solutions tailored to meet the specific needs of its clients across various sectors.

1.1 Background of the Company

AddisWay Technology Solutions was founded to address the growing demand for digital transformation in Ethiopia and surrounding regions. The company has since become a trusted partner for organizations seeking reliable and effective technology solutions. It primarily offers:

- Main Products and Services: AddisWay provides custom software development, IT consulting, digital transformation services, and network infrastructure management.

- Main Customers and End Users: The company serves a wide array of clients, including small to medium enterprises (SMEs), corporations, government agencies, educational institutions, and NGOs. End users of its products and services include business professionals, IT managers, educators, and government employees.
- Organizational Structure: The company operates under a collaborative structure that includes software development, consulting, project management, and client support teams. Each team works closely with clients to understand their unique needs and deliver customized solutions.
- Workflows: AddisWay follows an Agile workflow, which allows for iterative development and regular client feedback. This includes sprint-based task organization, frequent progress reviews, and a focus on delivering high-quality, scalable solutions that align with client objectives.

1.2 Mission of the Company

AddisWay's mission is to enable clients to thrive in the digital era by delivering high-quality technology solutions that enhance operational efficiency, growth, and competitiveness. The company strives to create sustainable, user-friendly products that meet the unique needs of its clients.

1.3 Vision of the Company

The vision of AddisWay is to become a leading technology partner in Ethiopia and across Africa, setting new standards in software development, IT consulting, and digital transformation services. The company aims to continuously expand its expertise and build long-term partnerships to foster innovation in the region.

1.4 Goal of the Company

The goals of AddisWay Technology Solutions include delivering high-quality, reliable technology solutions, staying at the forefront of technological advancements, supporting the digital transformation of businesses, and fostering a collaborative and growth-oriented work culture.

1.5 Main Products and Services of the Company

AddisWay Technology Solutions offers a range of products and services to support the digital needs of modern organizations:

- Custom Software Development: Building tailored applications that meet specific client requirements.
- IT Consulting: Providing guidance on IT strategy, infrastructure optimization, cybersecurity, and project management.
- Digital Transformation Solutions: Implementing advanced technologies to optimize business processes, including data analytics and cloud solutions.
- Network and Infrastructure Services: Design, installation, and management of secure IT infrastructures.

The purpose of my internship at AddisWay was to contribute to their ongoing projects, specifically by assisting in software development tasks that required a thorough understanding of modern web technologies, while also gaining hands-on experience in a professional setting.

1.6 Main Customers and End Users of the Company

AddisWay's main customers include:

- Small to Medium Enterprises (SMEs), who need digital tools to enhance productivity and remain competitive.
- Corporations, which rely on custom software solutions and IT consulting to improve efficiency.
- Government Agencies, which require secure and efficient digital transformation services.
- Educational Institutions, benefiting from software for administrative and academic purposes.
- Non-Governmental Organizations (NGOs), which use digital tools to improve communication and data management.

End users include business professionals, IT administrators, educators, public sector employees, and others who rely on these solutions for improved efficiency and productivity.

2. Overall Internship Description and Activities

During my internship at AddisWay Technology Solutions, my role was within the software engineering department, focusing on web development using the MERN stack. The purpose of the internship was to gain hands-on experience in developing real-world applications and to apply my academic knowledge to practical projects. My responsibilities involved working on various phases of software development, from design and coding to testing and debugging.

2.1 Objectives of the Internship

The objectives of my internship were closely aligned with my academic background in software engineering and my career goal of becoming a skilled full-stack developer.

2.1.1 General Objective

The general objective of the internship was to gain practical experience in software development, particularly in web application development using the MERN stack. This included developing skills in front-end and back-end coding, as well as understanding how to build scalable applications.

2.1.2 Specific Objectives

The specific objectives of the internship included:

1. Learning Agile Development Practices: Understanding and participating in Agile development processes, including sprint planning, task tracking, and code reviews.
2. Building Proficiency in the MERN Stack: Developing and enhancing technical skills in MongoDB, Express, React, and Node.js.
3. Gaining Real-World Problem-Solving Experience: Applying theoretical knowledge to identify and solve practical software engineering challenges.
4. Collaborating on Software Projects: Working closely with team members to contribute to ongoing projects and improve interpersonal communication skills.
5. Understanding Client Requirements: Learning to gather, interpret, and implement client requirements effectively within the project scope

2.2 Nature of work/Tasks carried out

1. **Week One**
   I focused on building a solid foundation in React. I learned about React's component-based architecture, JSX, and the virtual DOM, which are essential for creating efficient, interactive user interfaces. By the end of the week, I had a good grasp of using the useState hook for managing state in components, which was reinforced through hands-on challenges.

2. **Week Two**

involved applying my React skills to start building the Complaint Management System. I focused on creating UI components like complaint submission forms and managing user inputs effectively. The challenges helped integrate these components into a functional interface, making significant progress in the frontend development.

3. **Week Three**
   In Week 3, I began learning Node.js and its integration with React. I set up a basic Node.js server and created simple API endpoints using Express. These endpoints enabled data exchange between the React frontend and the Node.js backend, marking a key step in the project's development.

4. **Week Four**
   Week 4 was dedicated to enhancing the Complaint Management System using advanced React techniques. I implemented global state management with the context API and refined the UI. The challenges involved managing asynchronous operations and integrating the frontend with the backend more effectively.

5. **Week Five**
   Week 5 focused on using AI for content creation. I explored various AI tools for generating text and media, learning how they can streamline creative processes. The weekend challenge involved using AI to create and integrate content into a project, showcasing AI's practical applications.

6. **Week Six**
   In Week 6, I learned about AI assistants and their integration into various platforms. I built a simple AI assistant using available tools and APIs. The weekend challenge involved creating a chatbot, which enhanced my understanding of AI's role in automating user interactions.

7. **Week Seven**
   explored AI in decision-making. I learned about tools that analyze data and predict outcomes, aiding informed decisions. The weekend challenge applied these tools to a decision-making scenario, highlighting AI's potential in improving business and project management processes.

8. **Week Eight**
   The final week focused on using AI to drive business innovation. I explored AI technologies for automating processes and enhancing customer experiences. The weekend challenge involved creating a business proposal integrating AI solutions, offering insights into AI's transformative impact on business operations.

3. Experience and Transferrable Skills

3.1 Work Experience

3.1.1 Technical Experience

During my internship at AddisWay Technology Solutions P.L.C., I worked extensively on developing web applications using the MERN stack (MongoDB, Express, React, and Node.js). My technical experience included:

- Frontend Development with React: I gained hands-on experience in building user interfaces using React. This involved creating reusable components, managing state using React hooks, and optimizing the application for better performance.
- Backend Development with Node.js and Express: I developed server-side logic using Node.js and Express to handle API requests and manage data flow. I worked with RESTful APIs, allowing for efficient data communication between the client and server.
- Database Management with MongoDB: I learned how to design and manage databases using MongoDB, including creating schemas, managing relationships, and performing CRUD operations. This experience provided insight into handling unstructured data in a NoSQL database environment.
- Version Control with Git: I used Git for version control, collaborating with team members and managing code changes effectively. This improved my understanding of Git workflows, including branching, merging, and resolving conflicts.

3.1.2 Non-Technical Experience

In addition to technical skills, I gained valuable non-technical experience, including:

- Team Collaboration: I worked as part of a team, which involved regular discussions, code reviews, and collaborative problem-solving. This taught me the importance of clear communication and alignment with team objectives.
- Project Planning and Time Management: Working on projects with specific deadlines required me to prioritize tasks, manage my time efficiently, and adapt to changes in project requirements.
- Client Communication: I learned how to interpret client requirements and communicate progress effectively. Understanding client needs and expectations was crucial to delivering a product that met project goals.

3.2 Benefits of the Experience

Throughout the internship, I developed and enhanced several skills that contributed to my professional growth:

- Technical Skills: I strengthened my knowledge of the MERN stack and applied it to real-world scenarios, building upon the theoretical knowledge I gained from my coursework. This experience helped me understand how to structure and organize code, debug issues, and create scalable applications.
- Soft Skills: My teamwork, communication, and problem-solving skills improved significantly. Working closely with team members on projects and participating in team discussions helped me become more adaptable and open to different perspectives. Regular feedback sessions allowed me to learn from mistakes and continuously improve my approach.

3.3 Skills Gained from Experience

- Practical Skills:
    - Frontend Development: I became proficient in building web interfaces with React, learning to create dynamic, responsive, and user-friendly applications.
    - Backend Development: I developed skills in handling server logic, managing APIs, and organizing data flow using Node.js and Express.
    - Database Management: I learned effective techniques for managing NoSQL databases, including schema design, query optimization, and data storage strategies.

- Theoretical Knowledge:
    - Application of Coursework: I applied concepts learned in my software engineering and database management courses, such as CRUD operations, RESTful API principles, and agile development methodologies.
    - Understanding of Web Application Architecture: The internship gave me practical insights into the architecture and flow of full-stack applications, from user requests to data processing and storage.
- Team-Playing Skills:
    - Collaborative Coding: I gained experience working on shared codebases, participating in code reviews, and following team workflows, which emphasized the importance of clean, organized code for easier collaboration.
    - Effective Communication: Regular team meetings and status updates improved my ability to communicate project progress, challenges, and solutions with team members effectively.

- Leadership Skills:
    - Initiative and Responsibility: I took ownership of tasks within my project, often suggesting solutions for problems and proactively seeking help when needed. This helped me develop a sense of responsibility and initiative in my work.
    - Peer Support: Occasionally, I helped other interns with technical issues, which improved my mentoring skills and reinforced my understanding of the technologies we used.

3.4 Challenges Faced and Solutions Implemented

Throughout the internship, I encountered a few challenges that provided valuable learning experiences.

1. Technical Challenges
    - Challenge: Debugging complex bugs and ensuring seamless integration between the frontend and backend.
    - Solution: I addressed this by dedicating time to thoroughly review documentation, perform step-by-step debugging, and consult with team members when necessary.

2. Collaboration Challenges
    - Challenge: Initially, coordinating with team members working in different areas (frontend vs. backend) was challenging.
    - Solution: We scheduled regular meetings to discuss our progress, align our tasks, and address issues in real-time, improving overall collaboration.

3. Time Management
    - Challenge: Balancing multiple tasks within the project while meeting deadlines was sometimes difficult.
    - Solution: I adopted a task prioritization system, breaking down tasks into smaller, manageable parts and allocating specific times to complete each.

4. Personal Growth Challenges
    - Challenge: Staying motivated and continuously learning new concepts in a fast-paced environment.
    - Solution: I kept track of my progress to motivate myself and engaged in self-paced learning resources to keep up with new tools and practices in the MERN stack.

Part Two: Real-World Project Documentation

4. General Overview of the Project

4.1 Background of the Project

The Complaint Management System (CMS) project was initiated to address the need for an efficient and structured approach to managing complaints within an organization. By providing employees with a straightforward way to submit complaints and enabling departments to track and resolve these issues, the CMS improves organizational transparency, compliance, and employee satisfaction. This system leverages the MERN stack (MongoDB, Express.js, React, and Node.js) for robust, scalable, and interactive web applications. The CMS also offers administrators and department heads specialized tools to oversee complaints, assign responsibilities, track resolutions, and ensure adherence to compliance standards.

In an organizational environment, complaints and grievances can go unnoticed or unresolved without a systematic way to manage them. The CMS aims to fill this gap by automating complaint handling, establishing a clear workflow from complaint submission to resolution, and creating accountability. This project not only improves the user experience for employees but also helps the organization meet internal and external compliance requirements effectively.

4.2 Objective of the Project

4.2.1 General Objective

The general objective of the CMS project is to create a user-friendly, scalable, and secure web-based system that enables employees to submit complaints, allows department heads to manage complaints relevant to their area, and provides administrators with oversight and tracking capabilities. This objective is driven by the need to create a centralized platform that improves compliance, accountability, and efficiency in addressing employee complaints.

4.2.2 Specific Objectives

To achieve the overarching goal, the CMS project focuses on several specific objectives:

1. Complaint Submission Module: Develop a simple and accessible interface for users to submit complaints, categorize issues by type and severity, and attach relevant documentation.
2. Automated Routing and Notifications: Implement automated routing of complaints to appropriate departments and individuals based on predefined criteria, as well as a notification system to inform stakeholders of new or escalated issues.

3. Admin and Department Panel: Provide an administrative dashboard for tracking complaints by category, department, and severity, and enable department heads to manage and resolve complaints relevant to their area.
4. Complaint Status Tracking: Develop features that allow users and administrators to monitor complaint status updates in real time and ensure complaints are addressed within the necessary timelines.
5. Knowledge Base: Develop a knowledge base for recurring issues to expedite complaint handling in the future.

4.3 Statement of the Problem for the Project

Organizations often struggle with handling and tracking employee complaints effectively. Without a structured complaint management system, complaints may go unresolved, leading to compliance risks, poor employee satisfaction, and potential legal challenges. The CMS project addresses these issues by providing a systematic approach for complaint submission, categorization, assignment, and resolution. By integrating automated notifications, case tracking, and a feedback system, the CMS enhances both employee satisfaction and organizational compliance, enabling timely and transparent resolution of complaints.

Key problems addressed by the CMS include:

1. Lack of a centralized system for complaint submission and tracking.
2. Delayed or missed resolutions due to manual complaint handling.
3. Insufficient communication and transparency between employees and management on complaint status.
4. Absence of a feedback loop to evaluate and improve complaint handling processes.

4.4 Methodology

The CMS project was developed using the Agile methodology, which facilitated a flexible and iterative approach. This allowed for incremental progress, feedback integration, and continuous improvement throughout the project.

1. Requirement Gathering: Conducted meetings with stakeholders to understand complaint management needs, specific requirements for user roles (User, Admin, Department Head), and compliance standards.
2. System Design and Prototyping: Designed the system architecture, including front-end and back-end components, and created prototypes to visualize the workflow.
3. Development Cycles (Sprints): Divided the project into smaller sprints, with each sprint focused on implementing specific modules or functionalities. For instance,

one sprint was dedicated to creating the complaint submission form, another focused on building the admin dashboard, and so on.

4. Testing and Validation: Conducted thorough testing after each sprint to validate feature functionality, identify bugs, and refine components based on test results.
5. Feedback and Iteration: After initial testing, collected feedback from potential end-users to make improvements, especially in user interface design and usability.
6. Final Integration and Deployment: Combined all developed modules, conducted end-to-end testing, and prepared the system for deployment on a web server.

4.5 Development Tools

The CMS project leveraged the MERN stack for its flexibility, scalability, and ease of development. Each tool in the stack played a crucial role in building the system.

1. **MongoDB:** A NoSQL database used for storing complaint records, user information, and department structures. Its schema-less nature is well-suited for managing varied and complex complaint data.
2. **Express.js**: A backend web application framework for Node.js that simplifies API creation, handling data routing, and managing middleware functions in a structured way.
3. **React:** A JavaScript library used to build the interactive user interface, including complaint submission forms, real-time notifications, and an administrative dashboard.
4. **Node.js:** Used to manage server operations, handle API requests, and run asynchronous operations, enabling efficient processing of complaint submissions, real-time updates, and notification triggers.

- Additional Tools:
    - **Git:** Version control for tracking code changes and enabling collaborative development.
    - **Postman:** Used for testing API endpoints, ensuring that data is sent and received correctly.
    - **Trello**: Tools for project management and tracking sprint progress, task assignments, and development milestones.

4.6 Functional and Nonfunctional Requirements

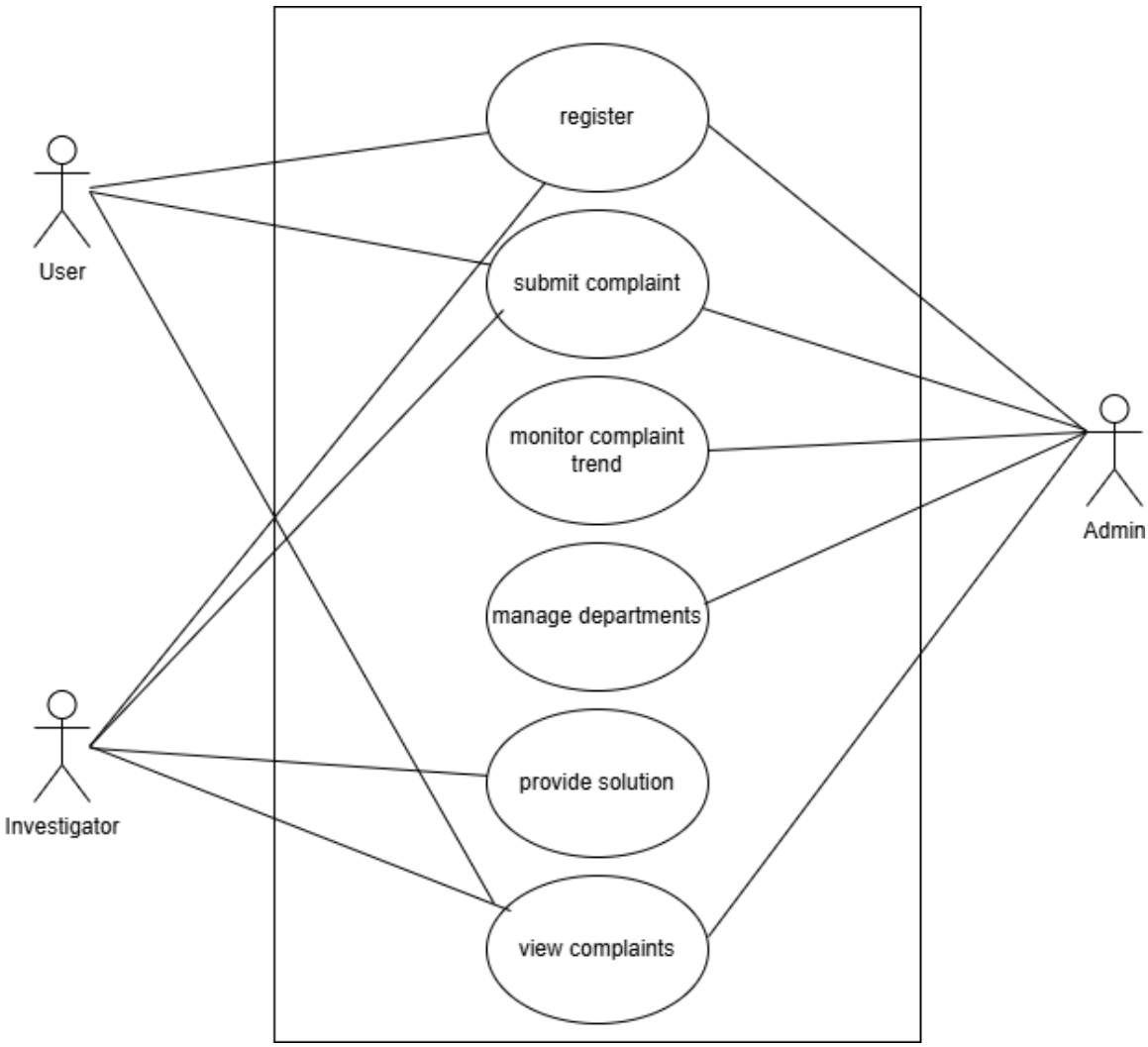
- Functional Requirements:
    - User Authentication: Secure login for different roles (employee, admin, department head).
    - Complaint Submission: Users can submit complaints with categorized details and file attachments.

- Automated Complaint Assignment: Complaints are routed to relevant departments based on type and severity.
- Notification System: Sends real-time notifications to involved parties about complaint status updates.
- Case Management: Track complaints, assign cases, and provide department heads with control over unresolved complaints.
- Feedback and Knowledge Base: Archive resolved complaints with solutions and gather feedback for future improvements.

- Nonfunctional Requirements:
  - Scalability: Capable of handling large volumes of complaints and users.
  - Security: Protect sensitive information through secure authentication and data encryption.
  - Usability: Ensure the system is intuitive, with a minimal learning curve for new users.
  - Reliability: Designed to maintain high availability and consistent performance.

4.7 Use Case, Class Diagram, and Sequence Diagram

1. **Use Case Diagram:** Outlines user interactions for complaint submission, complaint tracking, and complaint resolution, showing roles such as employees, admins, and department heads.

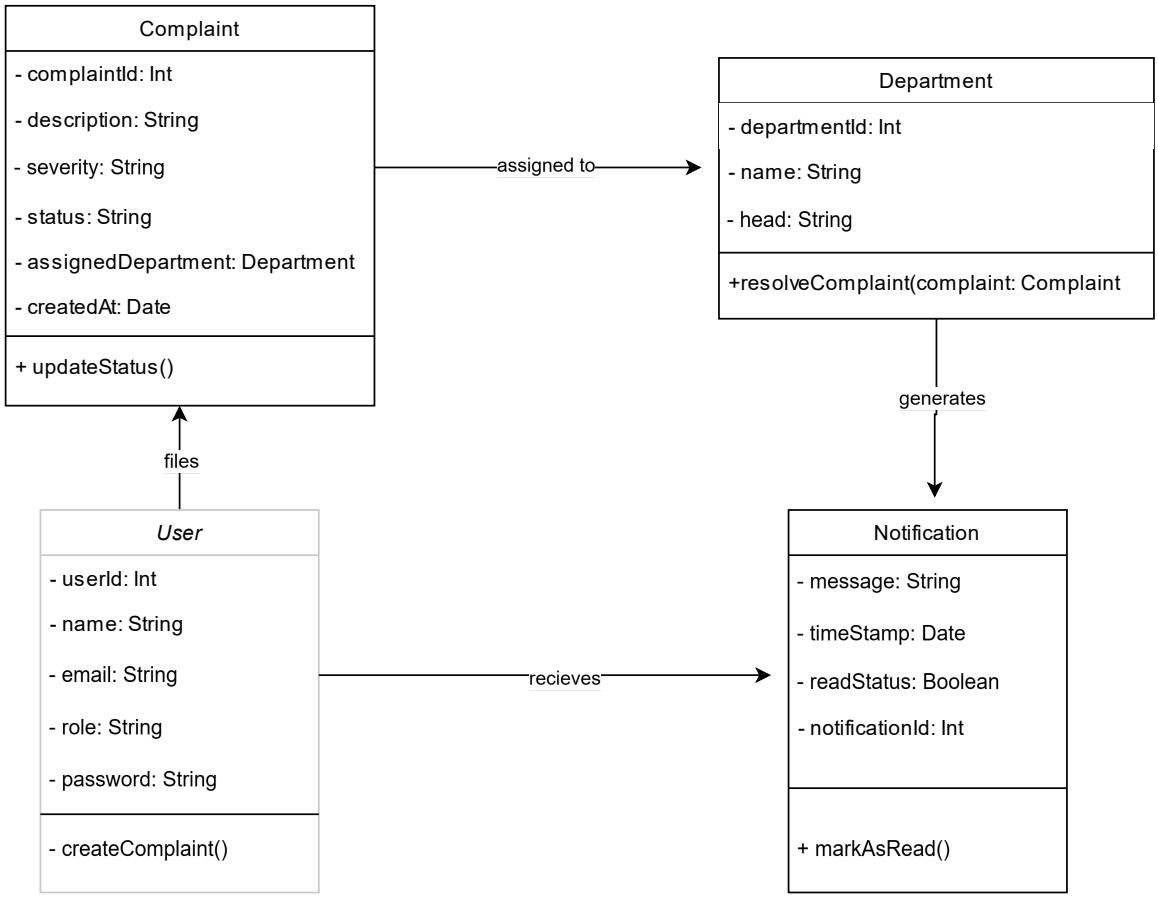


This diagram visualizes the roles and their interactions with the system's features.

Description:

- User: Submits complaints, views complaint status, and provides feedback.
- Admin: Manages users, monitors complaint trends, and resolves escalated complaints.
- Department Head: Oversees complaints relevant to their department, tracks progress, and provides solutions.

2. **Class Diagram:** Illustrates the main entities (User, Complaint, Department) and their relationships, specifying attributes like user roles, complaint types, and complaint status.



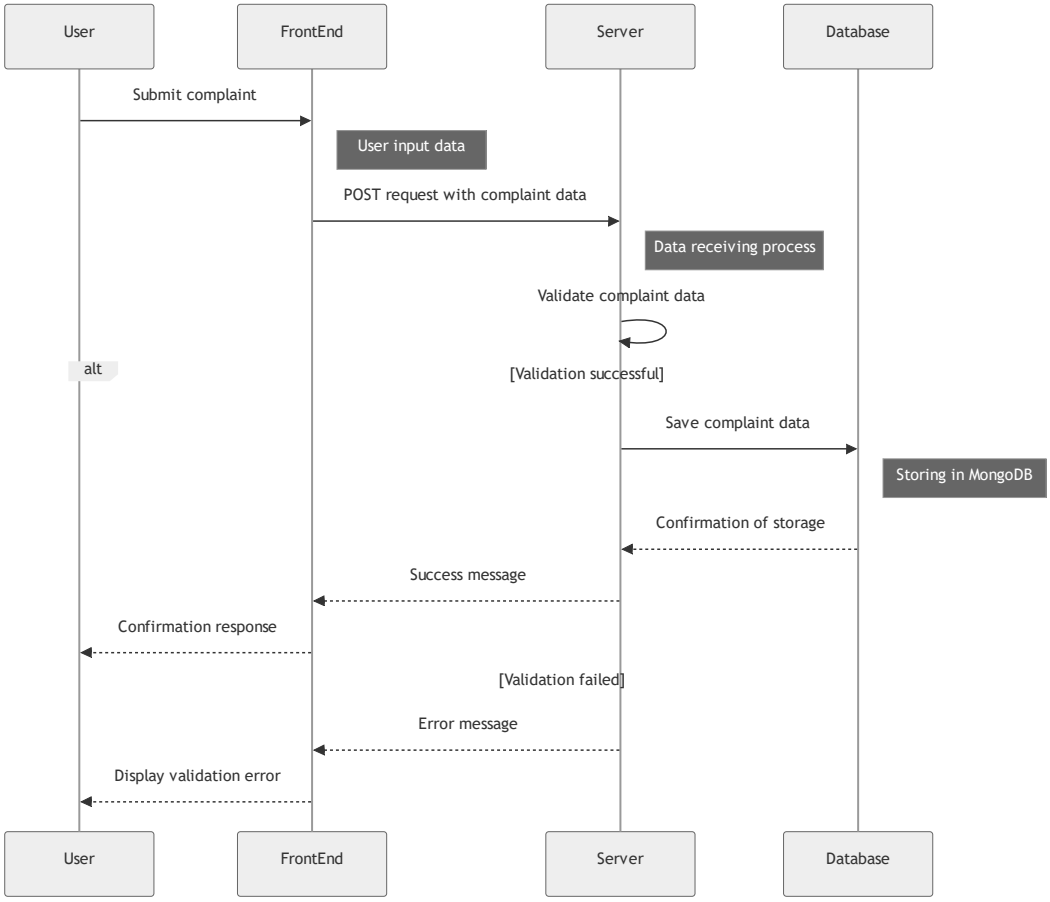
The class diagram provides a structured view of the system entities, their attributes, and relationships.

Main Classes:

- User: Attributes include userId, name, email, role, and password.
- Complaint: Attributes include complaintId, description, severity, status, assignedDepartment, and createdAt.
- Department: Attributes include departmentId, name, and head.
- Notification: Attributes include notificationId, message, timestamp, and readStatus.

3. **Sequence Diagram:** Shows the process flow from complaint submission to final resolution, detailing interactions between user interfaces, API endpoints, and the database.

The sequence diagram below demonstrates the flow of actions during a complaint submission:



Steps:

a. User submits a complaint through the front end.
b. Front-End sends a POST request with complaint data to the Server.
c. Server validates the data, saves it to MongoDB, and generates a confirmation response.

d. Database stores the complaint, and Server returns a success message to the User interface.

4.8 Database Design

The CMS database is built using MongoDB to support flexible, schema-less data storage. This NoSQL approach enables efficient handling of various data types, such as complaints, users, and departments. Below is a detailed design of the primary collections and their respective fields.

1. Users Collection

The Users collection stores user profile and authentication details, along with role-based access information. This collection supports different user roles: Admin, Department Head, and Employee.

- Fields:
  - _id (ObjectId): Unique identifier for each user.
  - firstName (String): First name of the user.
  - lastName (String): Last name of the user.
  - email (String): User's email, also used for login.
  - password (String): Hashed password for authentication.
  - role (String): Defines user roles with values like "admin", "head", or "employee".
  - createdAt (Date): Timestamp when the user was created.

- Indexes:
  - Unique index on email to ensure each email is unique in the system.

- Relationships:
  - Linked to the Departments collection (for department heads).

2. Departments Collection

The Departments collection stores details about each department, enabling complaints to be routed to the appropriate department and providing department heads with access to their relevant complaints.

- Fields:
    - _id (ObjectId): Unique identifier for each department.
    - name (String): Name of the department (e.g., “HR”, “IT”).
    - head (ObjectId): References the user who is the head of the department, linking to the Users collection.
    - createdAt (Date): Timestamp when the department was created.

- Relationships:
    - The head field references a User document, representing the department head.

3. Complaints Collection

The Complaints collection is central to the CMS and stores details for each complaint submitted by employees. Each complaint is categorized, routed, and tracked through its lifecycle from submission to resolution.

- Fields:
    - _id (ObjectId): Unique identifier for each complaint.
    - description (String): Detailed description of the complaint.
    - severity (String): Specifies the severity level (e.g., “Low”, “Medium”, “High”).
    - status (String): Tracks the complaint status (e.g., “Unresolved”, “In Progress”, “Resolved”).
    - department (ObjectId): References the department to which the complaint is assigned.
    - reporter (ObjectId): References the user who submitted the complaint, linking to the Users collection.
    - createdAt (Date): Timestamp when the complaint was submitted.
    - updatedAt (Date): Timestamp for the most recent update.
    - attachments (Array of Strings): URLs or paths to files attached to the complaint, allowing users to submit supporting documents.
    - solution (String): Text field for recording the solution or response to the complaint, filled by the department head or admin.

- Relationships:
    - department references the relevant department from the Departments collection.
    - reporter links to the user who submitted the complaint from the Users collection.

4. Notifications Collection

The Notifications collection manages real-time alerts and updates sent to users and administrators regarding complaint status changes, new complaint submissions, and resolutions.

- Fields:
  - _id (ObjectId): Unique identifier for each notification.
  - userId (ObjectId): References the recipient of the notification, linking to the Users collection.
  - message (String): Text of the notification message.
  - complaintId (ObjectId): References the related complaint, linking to the Complaints collection.
  - readStatus (Boolean): Tracks whether the notification has been read.
  - timestamp (Date): Timestamp when the notification was created.

- Relationships:
  - userId references the user who should receive the notification.
  - complaintId links to the complaint that triggered the notification.

5. KnowledgeBase Collection

The KnowledgeBase collection stores resolved complaints and their solutions for future reference. This collection helps build a searchable repository for frequently encountered complaints, improving response times.

- Fields:
  - _id (ObjectId): Unique identifier for each knowledge base entry.
  - complaintId (ObjectId): References the original complaint, linking to the Complaints collection.
  - solution (String): Detailed solution provided for the complaint.
  - createdAt (Date): Timestamp when the solution was added to the knowledge base.

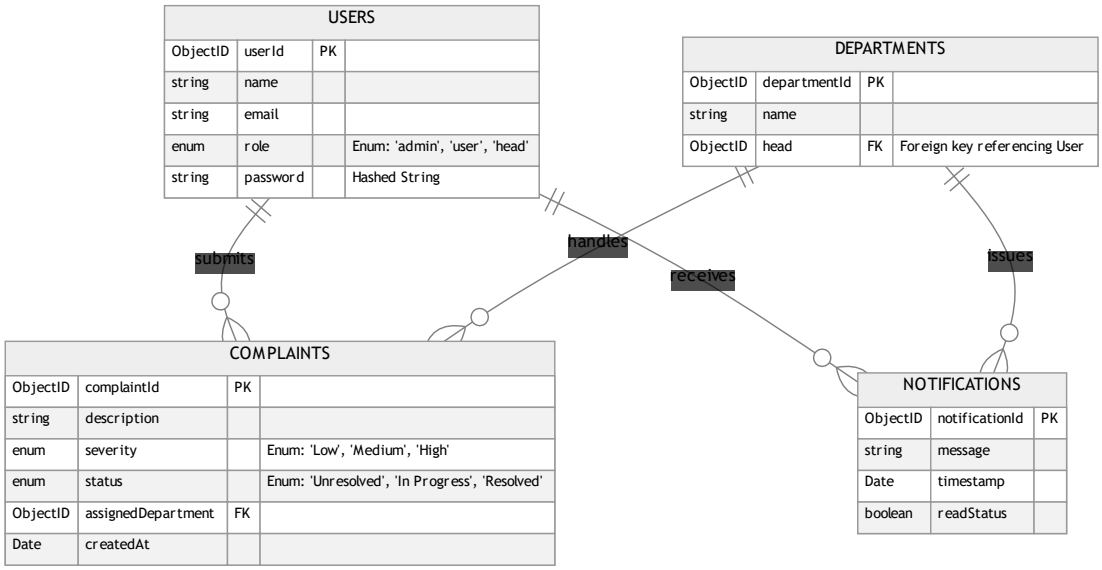
- Relationships:
  - complaintId links to the resolved complaint in the Complaints collection, allowing the complaint history to be accessed.

Entity-Relationship Diagram (ERD)

Here's how the relationships between these collections would appear visually in an ERD format:

Users have a one-to-many relationship with Complaints (each user can submit multiple complaints).

1. Departments have a one-to-many relationship with Complaints (each department may handle multiple complaints).
2. Complaints are referenced in Notifications and KnowledgeBase entries.
3. Users are referenced in Notifications as the recipient of each alert.



4.9 System Design

Component Diagram

The Component Diagram shows the major modules in the system and how they interact. It outlines the main components of the CMS and their responsibilities, as well as how they collaborate to achieve the project's objectives.

Core Components:

1. User Interface Module (React)
    - Description: Handles all user interactions and displays dynamic content, such as complaint submission forms, the admin dashboard, and notification panels.
    - Components:
        - Complaint Submission Form: Allows users to file complaints, select categories, and attach files.
        - Dashboard and Analytics: Displays complaint metrics, status counts, and trends in visual charts (using Recharts).
        - Notifications Panel: Displays updates for users regarding complaint statuses and actions taken by administrators or department heads.

2. Complaint Management Module (Express & Node.js)
    - Description: Manages complaint routing, case tracking, and complaint status updates.
    - Components:
        - Complaint Router: Receives complaint submissions from the User Interface Module, validates data, and forwards it to the database.
        - Case Tracker: Keeps track of complaint statuses (e.g., "Unresolved," "In Progress," "Resolved") and ensures administrators can view the progress.

3. Authentication and Authorization Module
    - Description: Manages user access and enforces role-based permissions.
    - Components:
        - User Login/Registration: Verifies user credentials, allows new user registration, and assigns roles.
        - Role-Based Access Control: Restricts access to specific features based on roles (e.g., Admin, Department Head, Employee).

4. Notification System
    - Description: Sends real-time alerts for complaint status updates and other relevant actions.
    - Components:
        - Notification Manager: Generates notifications for changes in complaint status, assignment, or escalation.
        - Message Queue (Optional): Used to handle notification delivery asynchronously for high traffic scenarios.

5. Knowledge Base Module

- Description: Archives resolved complaints along with their solutions for future reference and categorizes frequently occurring issues.
- Components:
  - Solution Archiver: Stores resolved complaints and their solutions for retrieval.
  - Search Engine: Allows users to search past complaints based on keywords or categories.

Interactions:

- The User Interface Module sends data to the Complaint Management Module for processing.
- The Notification System updates the UI with complaint status notifications.
- The Authentication Module restricts access to components based on user roles.

Deployment Diagram

The Deployment Diagram illustrates how the CMS components are deployed across different environments in a production setup. This includes details on server configurations, database hosting, and client-side deployment.

Deployment Architecture:

1. Client Side:
   - Technology: React
   - Deployment: Deployed on a CDN (Content Delivery Network) such as AWS CloudFront or Vercel to optimize loading speeds and enhance scalability.
   - Responsibilities: Renders the user interface, manages client-side routing, handles user interactions, and communicates with the server via RESTful API calls.

2. Application Server (Backend):
   - Technology: Node.js and Express.js
   - Deployment: Deployed on a cloud hosting platform such as Heroku, AWS EC2, or DigitalOcean.
   - Responsibilities:
     - Receives and processes API requests from the client side.
     - Manages business logic, such as routing complaints, handling escalations, and interacting with the database.
     - Ensures secure access to API endpoints through authentication and authorization.

- Environment Variables: Stores sensitive data (e.g., API keys, database URLs) in .env files for security.

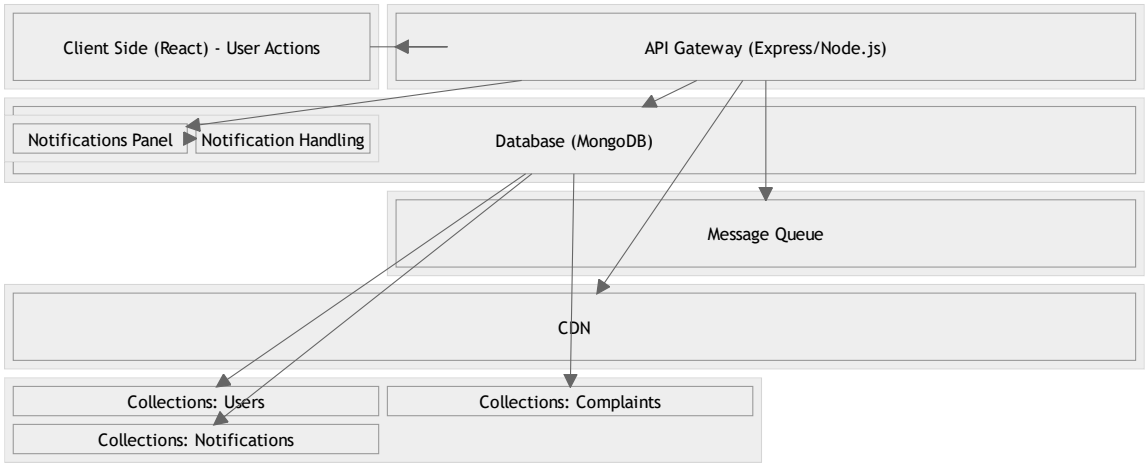
3. Database Server:
   - Technology: MongoDB
   - Deployment: Hosted on MongoDB Atlas or a similar managed cloud database service for ease of scalability and data security.
   - Responsibilities:
     - Stores and manages collections for Users, Complaints, Departments, Notifications, and Knowledge Base entries.
     - Handles queries for real-time data retrieval and supports data integrity across multiple requests.
     - Data Backup and Replication: Ensures high availability and durability of data through automated backup and replication features.

4. Optional Services:
   - Message Queue Service (for notifications): A service like Amazon SQS or RabbitMQ can be integrated to handle notification processing asynchronously, especially in high-traffic environments.
   - CDN: A CDN (e.g., AWS CloudFront, Cloudflare) is used for serving static assets and optimizing content delivery to end-users based on geographical location.

Block Diagram

Data Flow in the Block Diagram

1. User Actions: Users interact with the Client Side (React) to submit complaints, view statuses, and receive notifications.
2. API Requests: User actions generate API calls to the API Gateway (Express/Node.js), which processes these requests based on the role and action (e.g., complaint submission, complaint updates).
3. Database Interactions: The API Gateway interacts with the Database (MongoDB) for CRUD operations on collections like Users, Complaints, and Notifications.
4. Notification Flow: When a complaint status changes, the Notification Handling component sends updates to the Notifications Panel in the client interface. Optionally, a Message Queue can be used to manage high volumes of notifications.
5. Knowledge Base Access: Admins and users can access the Knowledge Base for solutions to past complaints, which the API Gateway retrieves from the database.



Part Three: Reflections and Recommendations

5.1 General Impressions about the Company

During my internship at AddisWay Technology Solutions P.L.C, I had a productive and enriching experience that provided hands-on exposure to real-world projects. The company has a positive work environment, where team members are supportive and willing to assist, which made it easier to adapt to new technologies and project requirements. The collaborative nature of the company culture fostered a learning-focused atmosphere, helping interns like myself develop technical skills while also gaining valuable insights into professional workflows.

The company's commitment to mentorship and guidance was especially valuable. Mentors were approachable, offering insights and assistance whenever needed, which allowed me to overcome challenges effectively without compromising my learning. This mentorship approach not only helped me grow as a developer but also instilled confidence in my problem-solving abilities. Overall, the company provides a welcoming and growth-oriented environment, which is ideal for interns and beginner developers.

5.2 Conclusion

In conclusion, my internship experience at AddisWay Technology Solutions P.L.C has been invaluable in developing both technical and professional skills. Working on real-world projects in a supportive, collaborative environment has helped me gain a practical understanding of the MERN stack, as well as experience with problem-solving in a team setting. The mentorship and guidance provided by the company were critical to my growth, allowing me to overcome challenges effectively and build a solid foundation for my future career in software development.

I highly recommend this company as an excellent learning environment for students interested in web development and software engineering. Future interns will find that AddisWay Technology Solutions P.L.C offers an ideal mix of practical experience, mentorship, and opportunities for personal and professional growth.

By following the recommendations outlined, future interns can maximize their learning experience, and the company can continue to provide an even more enriching environment for aspiring developers.

5.3 Recommendations

5.3.1 Recommendations for the Company

To further improve the experience for future interns, AddisWay Technology Solutions P.L.C could consider the following:

1. Technology Familiarization Period: Providing a brief onboarding period for interns focused on familiarizing them with the core technologies used in the projects would help reduce the learning curve and allow them to make productive contributions sooner.
2. Provide Documentation and Tutorials: Offering well-structured documentation, including project setup guides, tutorials, and best practices, would enable interns to navigate the codebase independently and expedite the onboarding process.
3. Encourage Pair Programming and Code Reviews: Pair programming sessions and regular code reviews would not only foster collaboration but also expose interns to effective debugging, code optimization, and teamwork strategies.
4. Allocate Time for Research and Learning: Allowing interns dedicated time for learning new tools or industry-standard practices relevant to the project could boost their confidence and adaptability.
5. Conduct Regular Check-Ins for Progress and Support: Structured check-ins with mentors would help interns stay on track with their goals, receive timely feedback, and feel more supported in their growth.

5.3.2 Recommendations for the Department

The department overseeing internships at AddisWay Technology Solutions P.L.C plays an essential role in preparing students for real-world experiences. Based on my experience, I suggest the following improvements to better equip students:

1. Pre-Internship Training on Core Technologies: Offering pre-internship workshops or short courses on the technologies commonly used in internships—such as JavaScript, databases, and basic web development—would provide students with a strong foundation, making it easier for them to adapt to industry requirements.
2. Guidance on Industry Expectations: Including a session to educate students on industry expectations, common workflows, and basic professionalism would better prepare them for the demands of an internship.
3. Encourage Practical Project Work: Prioritizing practical project assignments within coursework would enable students to build portfolios and gain hands-on experience before entering an internship, improving their confidence and job readiness.
4. Organize Peer Review and Presentation Sessions: Encouraging students to present their work and conduct peer reviews on projects would develop their collaboration and communication skills, making them more prepared for the collaborative nature of professional environments.

5.3.3 Recommendations for the University

The university can also play a significant role in enhancing students' internship experiences by providing additional support, resources, and guidance:

1. Expand Internship Placement Opportunities: Partnering with a broader range of companies and industries for internship placements would increase students' exposure to diverse fields, allowing them to explore various career paths and acquire a wider set of skills.
2. Organize Pre-Internship Workshops: Hosting workshops on essential technologies, industry practices, and professional skills would better equip students, reducing the initial adjustment period at the internship.
3. Provide Access to Learning Resources: Offering students access to professional online learning platforms (e.g., Coursera, Udemy) for supplementary study would help them stay updated with industry trends and gain a deeper understanding of the technologies used.
4. Arrange Follow-Up Sessions During Internships: By checking in with students during their internships, the university can provide additional support, address challenges, and gather feedback to improve future placements.
5. Facilitate an Internship Reflection Program: A structured reflection program after the internship—where students can discuss their experiences, challenges, and achievements—would help them articulate their learning outcomes and prepare for future opportunities.