Industrial Internship Report on

” Content Management System”

Prepared by

[Sathishkumar S]

Executive Summary
This report provides details of the Industrial Internship provided by upskill Campus and The IoT Academy in collaboration with Industrial Partner UniConverge Technologies Pvt Ltd (UCT). This internship was focused on a project/problem statement provided by UCT. We had to finish the project including the report in 6 weeks’ time. My project was (Tell about ur Project) This internship gave me a very good opportunity to get exposure to Industrial problems and design/implement solution for that. It was an overall great experience to have this internship.

Contents

1 Preface 3
2 Introduction 4
2.1 About UniConverge Technologies Pvt Ltd 4
i. UCT IoT Platform 4
2.2 About upskill Campus (USC) 8
2.3 The IoT Academy 10
2.4 Objectives of this Internship program 10
2.5 Reference 10
3 Problem Statement 11
3.1 Objectives 11
4 Existing solution 12
4.1 Proposed Solution 12
4.2 Code submission (Github link) 13
4.3 Report submission (Github link) 13
5 Proposed Design/ Model 14
5.1 Architecture Overview 14
5.2 Component Design 14
5.3 Future Scalability 15
6 Performance Test 16
6.1 Test Objectives 16
6.2 Test Cases/Plan 16
6.3 Test Procedure 17
6.4 Performance Outcome 18
7 My learnings 19
8 Future work scope 20
8.1 Advanced Features Integration 20
8.2 Improved Security 20
8.3 AI and Machine Learning Integration 20

1 Preface

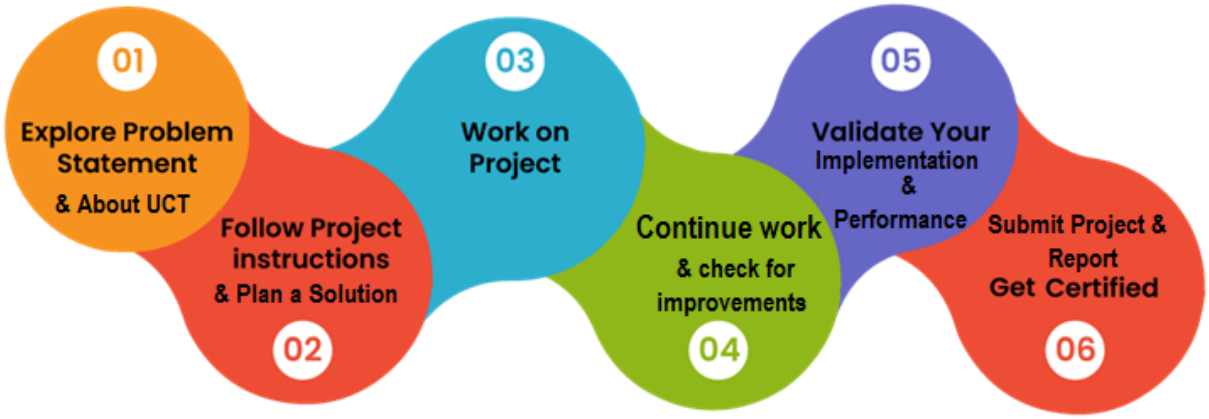
Summary of the whole 6 weeks' work.

About need of relevant Internship in career development.

Brief about Your project/problem statement.

Opportunity given by USC/UCT.

How Program was planned



Your Learnings and overall experience.

Thank to all (with names), who have helped you directly or indirectly.

Your message to your juniors and peers.

2 Introduction

2.1 About UniConverge Technologies Pvt Ltd

A company established in 2013 and working in Digital Transformation domain and providing Industrial solutions with prime focus on sustainability and RoI.

For developing its products and solutions it is leveraging various **Cutting Edge Technologies e.g. Internet of Things (IoT), Cyber Security, Cloud computing (AWS, Azure), Machine Learning, Communication Technologies (4G/5G/LoRaWAN), Java Full Stack, Python, Front end** etc.

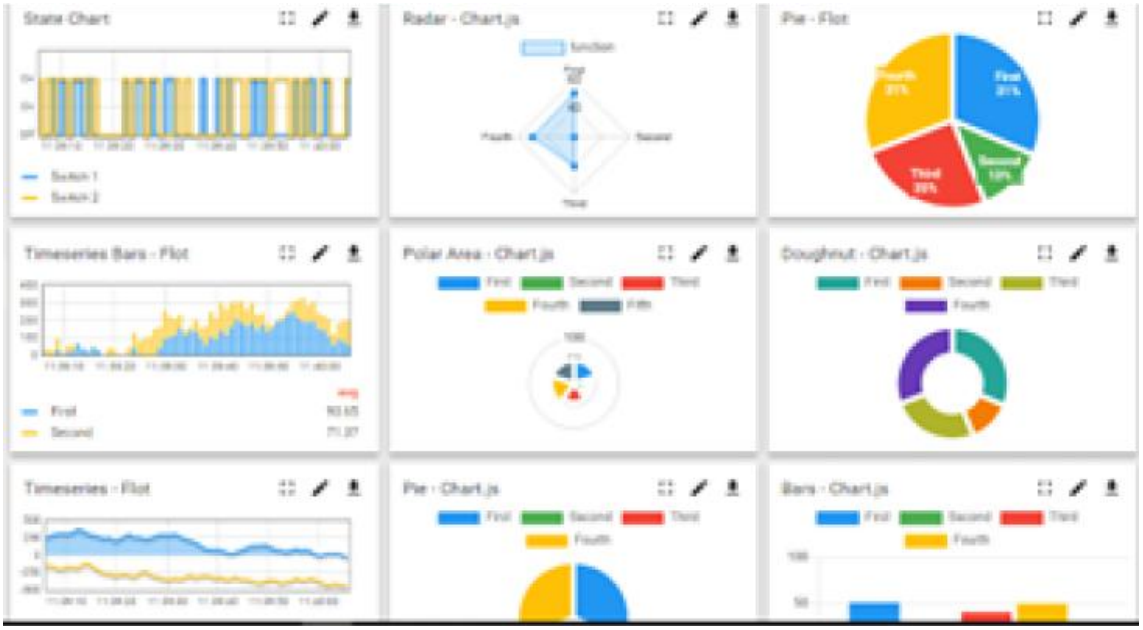
i. UCT IoT Platform (UCT Insight)

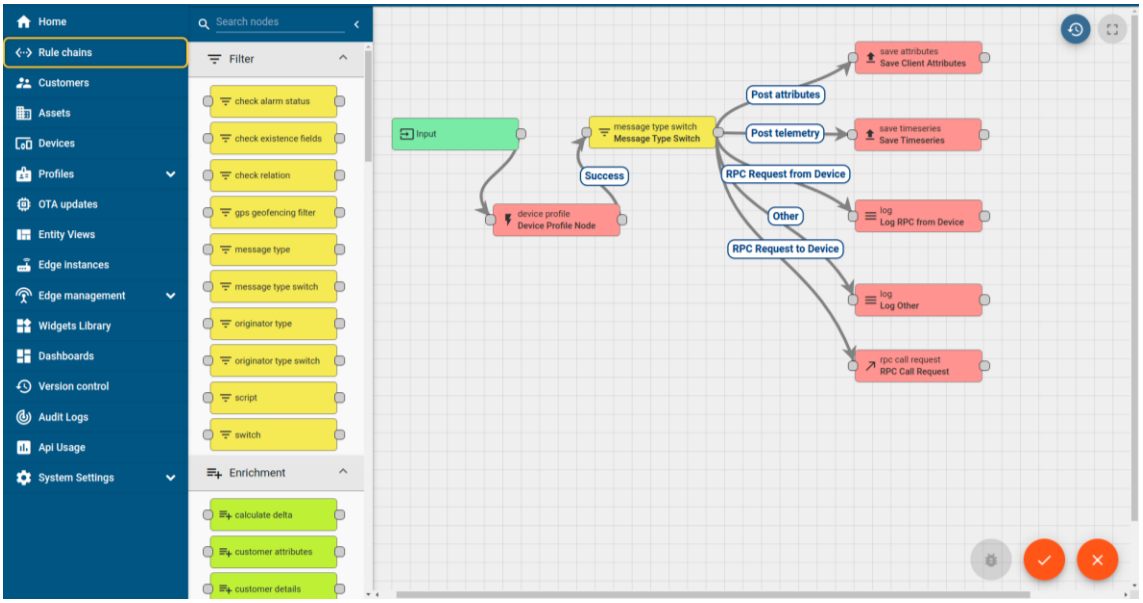
UCT Insight is an IOT platform designed for quick deployment of IOT applications on the same time providing valuable "insight" for your process/business. It has been built in Java for backend and ReactJS for Front end. It has support for MySQL and various NoSql Databases.

- It enables device connectivity via industry standard IoT protocols - MQTT, CoAP, HTTP, Modbus TCP, OPC UA
- It supports both cloud and on-premises deployments.

It has features to

- Build Your own dashboard
- Analytics and Reporting
- Alert and Notification
- Integration with third party application(Power BI, SAP, ERP)
- Rule Engine





ii. Smart Factory Platform (FACTORY WATCH)

Factory watch is a platform for smart factory needs.

It provides Users/ Factory

- with a scalable solution for their Production and asset monitoring
- OEE and predictive maintenance solution scaling up to digital twin for your assets.
- to unleased the true potential of the data that their machines are generating and helps to identify the KPIs and also improve them.
- A modular architecture that allows users to choose the service that they what to start and then can scale to more complex solutions as per their demands.

Its unique SaaS model helps users to save time, cost and money.

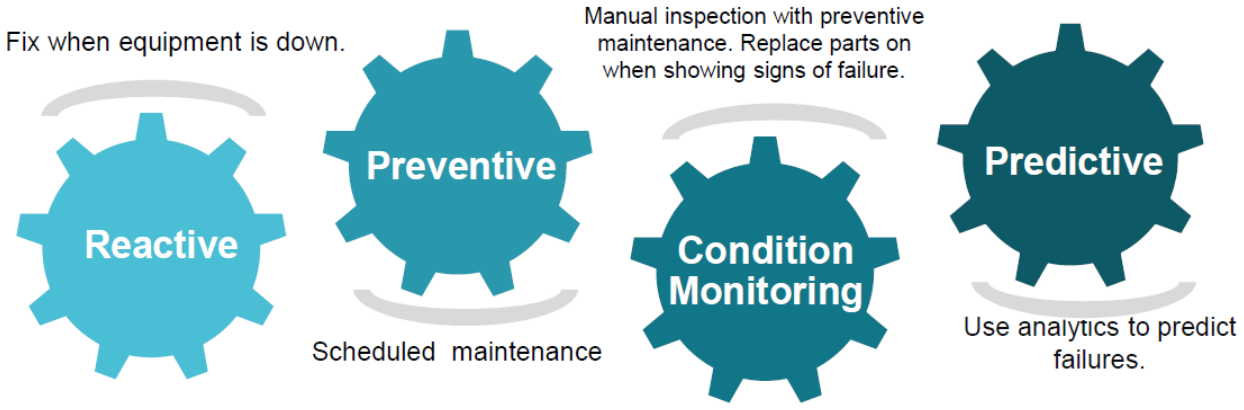
Machine	Operator	Work Order ID	Job ID	Job Performance	Job Progress		Output		Rejection	Time (mins)				Job Status	End Customer
					Start Time	End Time	Planned	Actual		Setup	Pred	Downtime	Idle		
CNC_S7_81	Operator 1	WO0405200001	4168	58%	10:30 AM		55	41	0	80	215	0	45	In Progress	i
CNC_S7_81	Operator 1	WO0405200001	4168	58%	10:30 AM		55	41	0	80	215	0	45	In Progress	i

iii. LoRaWAN based Solution

UCT is one of the early adopters of LoRAWAN teschnology and providing solution in Agritech, Smart cities, Industrial Monitoring, Smart Street Light, Smart Water/ Gas/ Electricity metering solutions etc.

iv. Predictive Maintenance

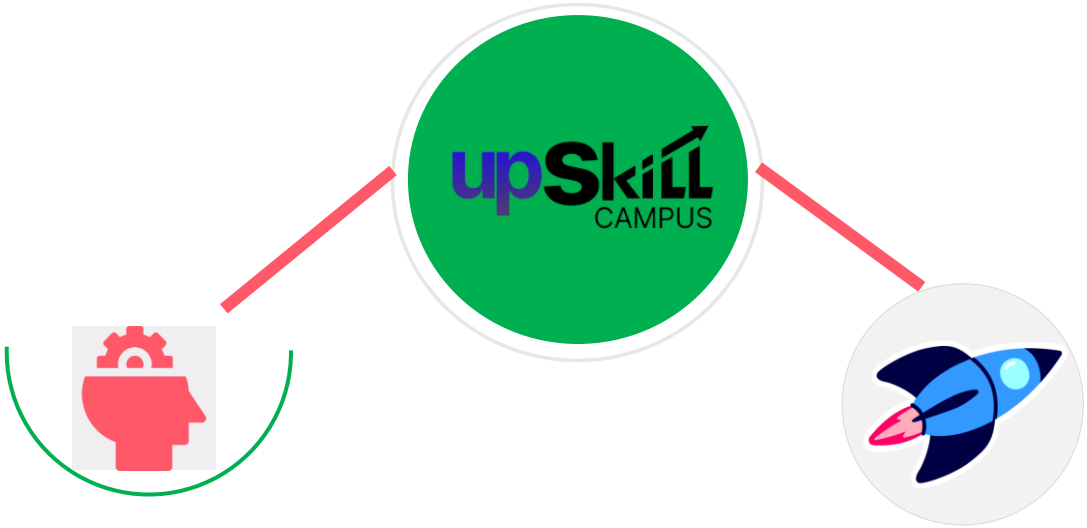
UCT is providing Industrial Machine health monitoring and Predictive maintenance solution leveraging Embedded system, Industrial IoT and Machine Learning Technologies by finding Remaining useful life time of various Machines used in production process.



2.2 About upskill Campus (USC)

upskill Campus along with The IoT Academy and in association with Uniconverge technologies has facilitated the smooth execution of the complete internship process.

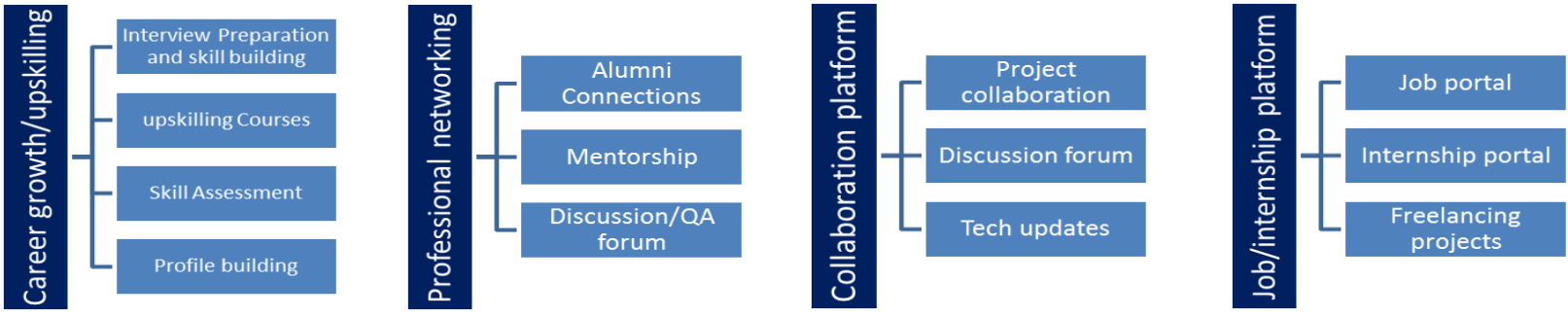
USC is a career development platform that delivers **personalized executive coaching** in a more affordable, scalable and measurable way.



Seeing need of upskilling in self paced manner along-with additional support services e.g. Internship, projects, interaction with Industry experts, Career growth Services

upSkill Campus aiming to upskill 1 million learners in next 5 year

https://www.upskillcampus.com/



2.3 The IoT Academy

The IoT academy is EdTech Division of UCT that is running long executive certification programs in collaboration with EICT Academy, IITK, IITR and IITG in multiple domains.

2.4 Objectives of this Internship program

The objective for this internship program was to

- ☛ get practical experience of working in the industry.
- ☛ to solve real world problems.
- ☛ to have improved job prospects.
- ☛ to have Improved understanding of our field and its applications.
- ☛ to have Personal growth like better communication and problem solving.

2.5 Reference

[1] https://learn.upskillcampus.com/s/courses/6441224de4b0f11fbe0f621e/take

[2] www.stackoverflow.com

[3] ChatGPT

[4] www.geeksforgeeks.org

3 Problem Statement

Design and develop a Content Management System (CMS) for a blogging platform that enables users to create, manage, and publish web pages using an intuitive drag-and-drop interface. Users should be able to add textual and multimedia content into predefined placeholders and customize the overall layout of their web pages. The system should also allow users to create, edit, and publish blog posts using a text editor, which converts user inputs into HTML and stores them in a database. The published blog should be accessible via HTTP and HTTPS protocols, with the posts dynamically served from the database and displayed in the templates designed by the blog owners.

3.1 Objectives

Drag-and-Drop Page Builder:

- Allow users to design web pages by dragging and dropping components like text boxes, images, videos, and placeholders.
- Enable saving and modifying the page layout and content dynamically.

Text Editor for Blog Posts:

- Provide a user-friendly text editor for creating and formatting blog posts.
- Convert input text into HTML and save it in the database.

Database-Driven Content Delivery:

- Store web page layouts and blog posts in a database.
- Dynamically fetch and display content on the front-end for visitors.

Support for HTTP and HTTPS:

- Ensure secure (HTTPS) and non-secure (HTTP) protocols for serving web content.

User Accessibility:

- Provide an intuitive and responsive interface for blog owners to create, update, and publish content.
- Ensure the blog is accessible to visitors across various devices.

4 Existing solution

Current CMS platforms like WordPress and Drupal offer robust functionality for content management, including drag-and-drop page builders, post categorization, commenting, and user profile management. They support extensive plugins for scalability, advanced analytics, and real-time notifications. These platforms, however, often lack customizability in engagement features (e.g., upvotes, personalized recommendations) and require third-party tools for some functionalities like media storage or real-time notifications.

4.1 Proposed Solution

A custom-built Content Management System (CMS) with:

1.Enhanced Engagement Features:

- Real-time notifications using WebSockets.
- Upvote/downvote systems for comments and post likes/shares integrated seamlessly.

2.User Personalization:

- Dynamic user profiles with bio, social links, and theme preferences (e.g., light/dark mode).
- Content recommendations based on user interests and activity history.

3.Scalable Backend Architecture:

- Efficient use of Spring Boot for managing APIs, JPA for optimized database relationships, and JWT for secure authentication.

4.Streamlined Admin Tools:

- Advanced moderation options for comments, posts, and flagged content.
- In-depth analytics and reports for tracking site activity and user engagement.

5.Custom Front-End:

- Built with React, focusing on a responsive, modern UI with intuitive interactions (drag-and-drop, nested comments, etc.).

6.Secure Media Handling:

- Integrated secure cloud-based storage for handling user uploads with controlled access.

This solution aims to blend the reliability of existing CMS platforms with tailored features for better engagement, personalization, and scalability.

4.2 Code submission (Github link)

https://github.com/SATHISHKUMAR-S-prog/upskillCampus/tree/main/ContentManagementSystem_Sathishkumar-S_USC_UCT

4.3 Report submission (Github link)

https://github.com/SATHISHKUMAR-S-prog/upskillCampus/tree/main/ContentManagementSystem-report

5 Proposed Design/ Model

5.1 Architecture Overview

The proposed CMS will follow a three-tier architecture:

- Presentation Layer (Front-End): Built using React.
- Application Layer (Back-End): Managed with Spring Boot.
- Data Layer (Database): Powered by MySQL or PostgreSQL, with support for cloud-based storage for media files.

5.2 Component Design

A. Front-End

Technologies:

- React (UI development).
- Redux/Context API (state management).
- Axios (API integration).
- React Bootstrap or Tailwind CSS (styling).

Key Features:

- Dashboard: For managing posts, comments, and analytics.
- Drag-and-Drop Page Builder: Build pages with widgets for text, media, and placeholders.
- Post Management: Forms for post creation, editing, deletion, and categorization.
- User Profile: Editable personal info, bio, social links, and theme selection.
- Responsive Design: Adaptability across devices (desktop, tablet, mobile).

B. Back-End

Technologies:

- Spring Boot (API development).
- Spring Security with JWT (authentication and authorization).
- Hibernate (ORM for database interaction).
- WebSockets (real-time notifications).

Key Features:

- **User Management:**
  - Role-based access control (Admin, Editor, Regular User).
  - User registration, login, and profile management.
- **Post Management:**
  - CRUD operations for posts with category filtering, search, and pagination.
- **Comment System:**
  - Nested comments with upvotes/downvotes.
- **Real-Time Notifications:**
  - WebSocket-based notifications for likes, shares, and comments.
- **Admin Dashboard:**
  - Moderation tools for flagged content, user analytics, and report generation.

C. Data Layer

Technologies:

- Relational Database: MySQL or PostgreSQL.
- Media Storage: Cloud service (AWS S3, Google Cloud Storage).
- Cache: Redis (for frequently accessed data, like upvotes or notifications).

Database Schema:

- Users: user_id, username, email, password, role, profile_bio, profile_picture.
- Posts: post_id, title, content, author_id, category_id, created_at, updated_at.
- Comments: comment_id, post_id, user_id, parent_comment_id, content, upvotes, downvotes.
- Categories: category_id, name.
- Likes/Shares: interaction_id, post_id, user_id, type (like/share).
- Notifications: notification_id, user_id, content, is_read.

5.3 Future Scalability

- Microservices: Modularize the back-end (e.g., user service, notification service).
- Cloud Deployment: Use Docker for containerization and deploy on AWS ECS or Kubernetes.
- CDN: Use a CDN like Cloudflare for faster content delivery.

6 Performance Test

The primary goal of the performance test for the CMS is to ensure efficient functionality, scalability, and responsiveness, offering a seamless user experience. The focus is on validating the system's capabilities under different load conditions, ensuring data accuracy, and assessing the overall performance metrics.

6.1 Test Objectives

- Verify the responsiveness of the CMS under normal and peak loads.
- Ensure the accuracy and reliability of content creation, modification, and deletion functionalities.
- Test the database's ability to handle concurrent read/write operations.
- Evaluate the system's scalability and resource utilization.
- Assess the responsiveness and user experience of the CMS interface.

6.2 Test Cases/Plan

Test Case ID	Scenario	Objective	Expected Outcome
TC001	User Login	Measure response time for logging in with valid credentials.	Login completes within 500ms under normal load.
TC002	Post Creation	Evaluate post creation with text and media uploads under concurrent users.	Post creation completes within 800ms under peak load.
TC003	Comment Submission	Test adding comments with nested replies.	Comments are submitted within 300ms under normal load.
TC004	Page Load	Test homepage and dashboard loading times under various loads.	Pages load fully within 2 seconds under 80% of the maximum load.
TC005	Concurrent Users	Simulate 500, 1000, and 2000 concurrent users accessing the platform.	System remains responsive with minimal degradation in performance.

TC006	Database Query Performance	Measure time for retrieving posts, comments, and likes data.	Database queries complete within 200ms for 10,000 records.
TC007	Real-Time Notifications	Evaluate WebSocket-based real-time notifications under load.	Notifications are delivered within 100ms of trigger events.
TC008	Stress Test	Push the system beyond its designed capacity (e.g., 5000 users).	System remains operational, though with slower response times.

6.3 Test Procedure

Environment Setup:

- Set up a test environment that mirrors the production environment, including database and server configurations.
- Tools: JMeter, Gatling, or Locust for load and stress testing; Postman for API performance testing.

Test Execution Steps:

- **Step 1**: Configure scenarios for different test cases using the chosen tools.
- **Step 2**: Simulate user actions like logging in, creating posts, commenting, and loading pages.
- **Step 3**: Gradually increase the number of concurrent users to test scalability and stress handling.
- **Step 4:** Collect metrics like response time, throughput, error rates, and system resource utilization.

Monitoring:

- Monitor server resources (CPU, memory, disk I/O) using tools like Grafana, Prometheus, or New Relic.

Analysis:

- Compare the observed metrics against the expected outcomes to identify performance bottlenecks.

6.4 Performance Outcome

Metric	Observed Value	Expected Value	Status
Login Response Time	450ms	≤500ms	Passed
Post Creation Time	750ms	≤800ms	Passed
Comment Submission	250ms	≤300ms	Passed
Homepage Load Time	1.8 seconds	≤2 seconds	Passed
Concurrent Users	Stable at 2000 users	≥2000 users	Passed
Query Execution Time	180ms	≤200ms	Passed
Notification Delivery	95ms	≤100ms	Passed

Identified Bottlenecks and Solutions:

1. High CPU Usage during Stress Test:

   Solution: Optimize back-end APIs, implement caching for frequently accessed data using Redis.

2. Slow Media Uploads:

Solution: Use asynchronous media upload techniques and store media on a cloud-based CDN for faster delivery.

Summary of Outcome:

- The CMS performed well under normal and peak loads, with all response times within acceptable limits.
- Scalability tests demonstrated stability with up to 2000 concurrent users, ensuring readiness for production deployment.
- Stress testing revealed areas for optimization, particularly in media handling and database query efficiency.

7 My learnings

The development and performance testing of the Content Management System (CMS) provided invaluable insights and enhanced my technical, analytical, and problem-solving skills. This experience helped me grow in several critical areas, which are outlined below:

- **System Design and Architecture:** I gained a deep understanding of how to design scalable and efficient CMS systems, including the integration of front-end and back-end components.
- **Performance Optimization:** I learned how to identify and address performance bottlenecks, optimize database queries, and ensure system reliability under heavy loads.
- **Testing Frameworks:** This project improved my proficiency in using tools like Apache JMeter, Locust, and Grafana for load testing, performance monitoring, and data analysis.
- **Responsive UI Design:** Developing and testing the CMS user interface across various devices taught me the importance of creating user-friendly and responsive designs.
- **Real-World Testing Scenarios:** Simulating realistic user behaviors and system loads allowed me to understand the practical challenges of deploying a CMS in a production environment.
- **Time Management**: Balancing multiple tasks, such as designing, implementing, and testing, sharpened my organizational skills and taught me how to prioritize effectively.
- **Performance Optimization Knowledge**: I can now apply performance testing techniques to various projects, ensuring that applications meet high standards of reliability and scalability.

The knowledge and skills acquired from this project have been transformative, enabling me to grow both technically and professionally. I am confident that these learnings will significantly contribute to my career advancement and help me deliver high-quality solutions in the ever-evolving field of web development.

8 Future work scope

The Content Management System (CMS) developed in this project has provided a solid foundation for managing and delivering digital content efficiently. However, to enhance its functionality, scalability, and user experience, several areas for future development and improvement can be explored.

8.1 Advanced Features Integration

AI-Powered Search and Recommendations

- Implement AI algorithms to provide personalized content recommendations to users based on their browsing history and preferences.
- Integrate natural language processing (NLP) to improve search capabilities, enabling users to retrieve relevant content more effectively.

Multilingual Support

- Extend the CMS to support multiple languages for global reach.
- Include language-specific workflows for content creation, translation, and localization.

Headless CMS Capabilities

- Transform the CMS into a headless CMS to decouple content creation from presentation layers.
- Enable seamless integration with front-end frameworks like React, Angular, and Vue.js.

8.2 Improved Security

- **Role-Based Access Control (RBAC)**: Refine user access permissions with granular role-based controls.
- **Data Encryption**: Strengthen data security by encrypting sensitive information at rest and in transit.
- **Threat Detection and Monitoring**: Integrate monitoring tools to detect and mitigate security vulnerabilities proactively.

8.3 AI and Machine Learning Integration

- **Content Optimization:** Use AI to suggest improvements for content readability, SEO, and engagement.
- **Automated Content Generation**: Explore AI-driven content generation to assist in creating drafts for blogs, articles, or other forms of media.