Industrial Internship Report on

Content Management System for a Blog

Prepared by

Kennice Gonsalves

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 *Content Management System for a Blog* which is a mini project that simulates the basic functionality of a platform like WordPress, allowing users to create, format, and "publish" blog posts. 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.

TABLE OF CONTENTS

1 Preface 3

2 Introduction 5

2.1 About UniConverge Technologies Pvt Ltd 5

2.2 About upskill Campus 8

2.3 Objective 10

2.4 Reference 10

2.5 Glossary 11

3 Problem Statement 12

4 Existing and Proposed solution 13

5 Proposed Design/ Model 15

5.1 High Level Diagram (if applicable) 17

5.2 Low Level Diagram (if applicable) 18

5.3 Interfaces (if applicable) 18

6 Performance Test 20

6.1 Test Plan/ Test Cases 21

6.2 Test Procedure 22

6.3 Performance Outcome 22

7 My learnings 23

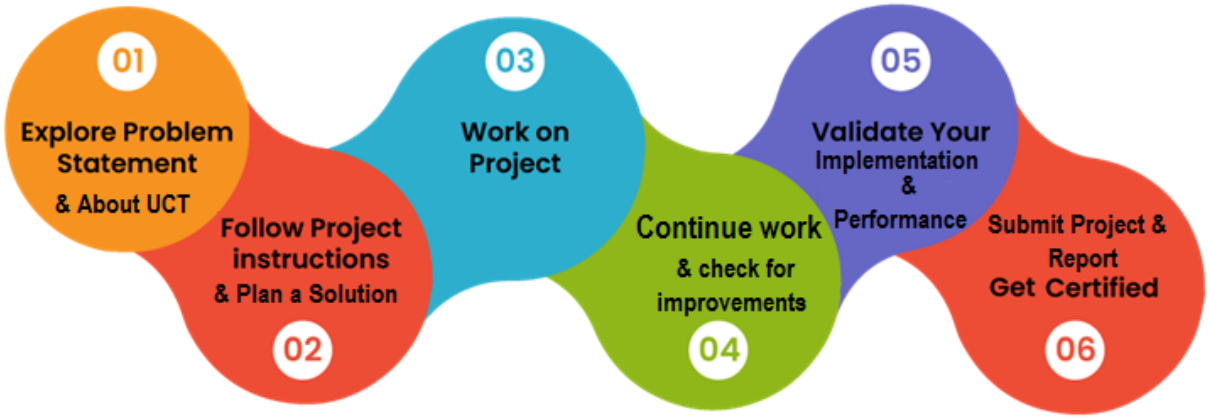
8 Future work scope 24

1 Preface

Summary of the whole 6 weeks' work:

1) About need of relevant Internship in career development.
2) Brief about Your project/problem statement.
3) Opportunity given by USC/UCT.

How Program was planned:



Your Learnings and overall experience:

Throughout this internship, I learned how to practically apply Java in a real-world-style project. Even though I initially felt underconfident and unsure about where to begin, this project helped me break out of that mental block and actually see my ideas take shape.

I gained confidence in:

- Writing backend logic with Java
- Simulating real backend processes like storage
- Structuring a full mini-project from scratch
- Troubleshooting setup issues and running Java locally via VS Code

- Managing stress under deadlines

This internship gave me a glimpse into what it's like to work with realistic project requirements and deliver something complete, even with limited tools and time.

Thanks to all (with names), who have helped you directly or indirectly:

I would like to sincerely thank Upskill Campus for providing this internship structure and helping me apply my academic skills to something practical.

I also want to thank my college, St. Francis Institute of Technology (SFIT), for creating a space where we can explore these projects and learn independently.

Your message to your juniors and peers:

To anyone who feels like they're not smart enough, or just lost in these placement chaos— You are not behind. You're just growing.

Start small. Ask questions. Use resources. Ask for help when you need it.
And never, ever let LinkedIn posts or comparison ruin your journey.

You don't need to be perfect—you just need to be willing.
Your skills will grow. Your confidence will catch up. Just keep building.

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.

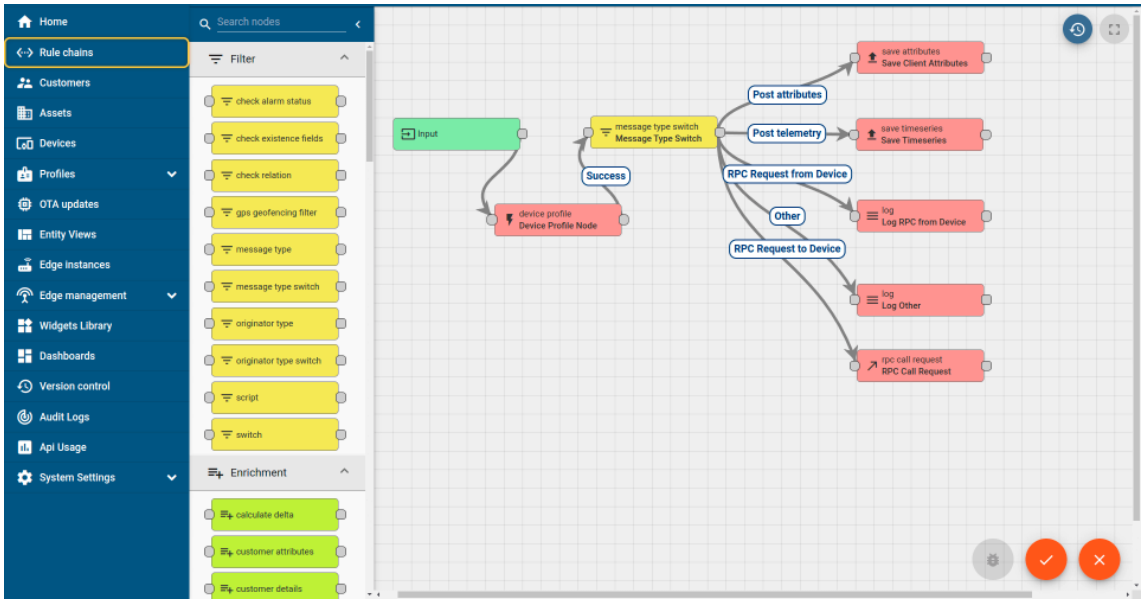
i. UCT IoT Platform (UCT Insight)

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 applications (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.

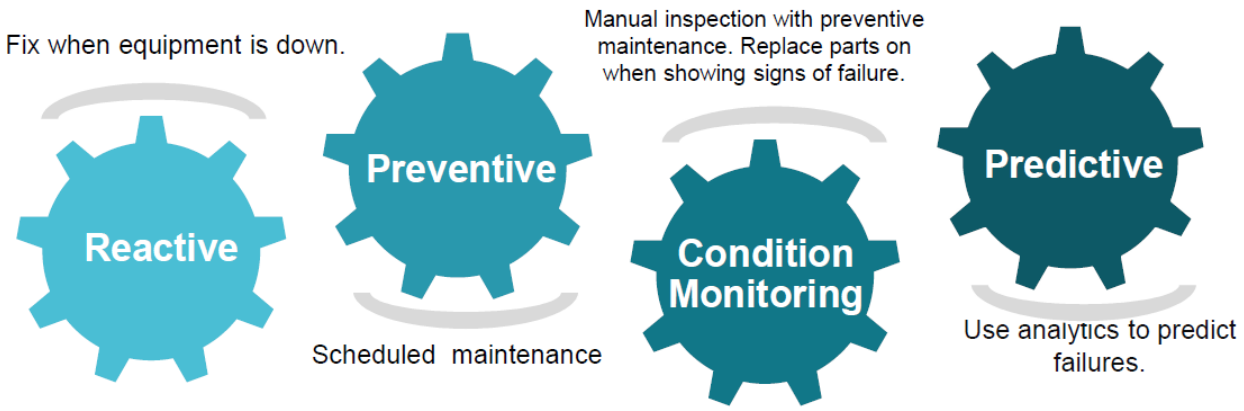
Machine	Operator	Work Order ID	Job ID	Job Performance	Job Progress		Output		Rejection	Time (mins)				Job Status	End Customer
					Start Time	End Time	Planned	Actual		Setup	Pred	Downtime	Idle		
CNC_S7_81	Operator 1	WO0405200001	4168	58%	10:30 AM		55	41	0	80	215	0	45	In Progress	i
CNC_S7_81	Operator 1	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 technology and providing solution in Aggrotech, Smart cities, Industrial Monitoring, Smart Street Light, Smart Water/ Gas/ Electricity metering solutions etc.

iv. Predictive Maintenance

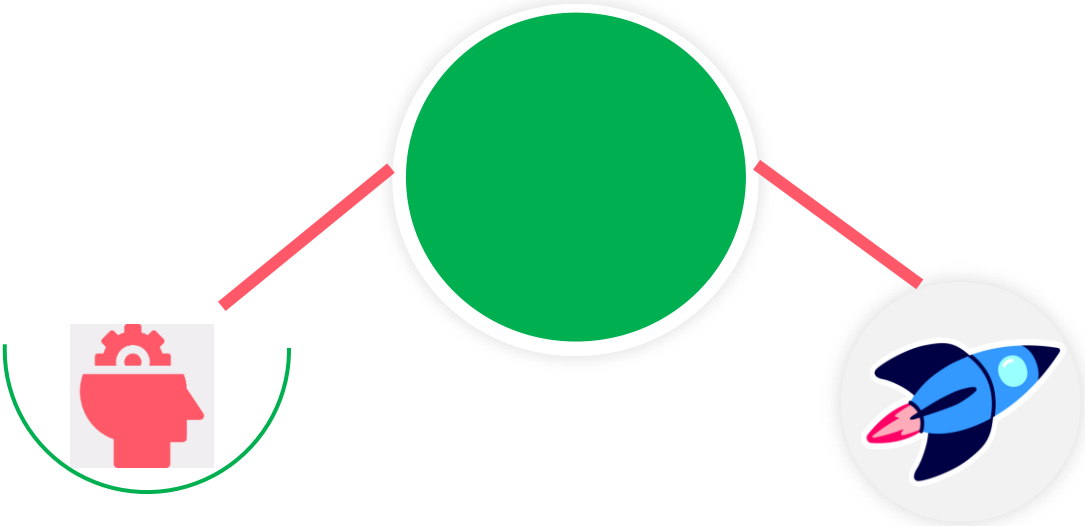
UCT is providing Industrial Machine health monitoring and Predictive maintenance solution leveraging Embedded system, Industrial IoT and Machine Learning Technologies by finding Remaining useful life time of various Machines used in production process.



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.

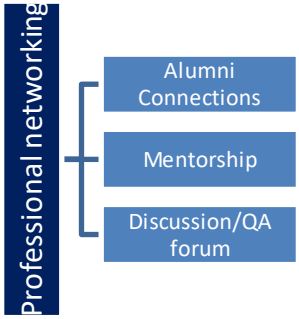
USC is a career development platform that delivers **personalized executive coaching** in a more affordable, scalable and measurable way.

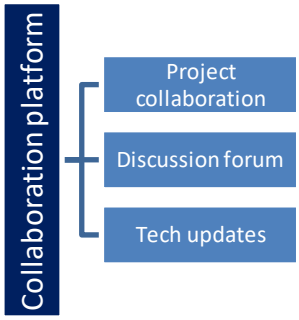


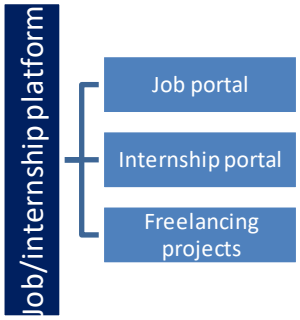
Seeing need of upskilling in self-paced manner along-with additional support services e.g. Internship, projects, interaction with Industry experts, Career growth Services

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

- ☛ 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 References

[1] UniConverge Technologies – Company Overview
https://uniconverge.com/

[2] UCT Insight IoT Platform – Product Page
https://uniconverge.com/uct-insight/

[3] Smart Factory Watch Platform – UCT
https://uniconverge.com/smart-factory-platform/

[4] The IoT Academy – EdTech Division
https://theiotacademy.co/

[5] Upskill Campus – Internship Partner Platform
https://upskillcampus.com/

[6] EICT Academy – IIT Kanpur
https://eict.iitk.ac.in/

2.6 Glossary

Acronym / Term	Full Form / Meaning
UCT	UniConverge Technologies Pvt. Ltd. – Digital transformation company
USC	Upskill Campus – Internship & skill development platform
IoT	Internet of Things – A network of interconnected smart devices
LoRaWAN	Long Range Wide Area Network – Wireless protocol used in IoT
OEE	Overall Equipment Effectiveness – Metric in smart manufacturing
SaaS	Software as a Service – Cloud-based software model
RUL	Remaining Useful Life – Predictive maintenance metric
EICT	Electronics & ICT Academy – Tech initiative from IITs
IITK/IITR/IITG	Indian Institutes of Technology (Kanpur/Roorkee/Guwahati)
CMS	Content Management System – Software to create/manage digital content
HTML	Hypertext Markup Language – Web page structuring language
VS Code	Visual Studio Code – Code editor developed by Microsoft
CLI	Command Line Interface – Text-based interaction with software
WYSIWYG	What You See Is What You Get – Visual editors without code
CRUD	Create, Read, Update, Delete – Basic operations on data

3 Problem Statement

In the assigned problem statements:

1) Content Management System for a blog

WordPress and Drupal would be the best examples of full stack web app development project ideas for students. Using the CMS users must be able to design a web page using the drag and drop method. Users should be able to add textual or media content into placeholders that are attached to locations on the web page using drag and drop method.

This way, users should be able to design the whole website. Users must also get an option to publish blog posts. For this, you need to have a text editor component that accepts user input text and converts it into HTML and push into a database.

The website must be published over HTTP and HTTPS protocols such that the blog posts are served from the database and displayed to the visitors in the page template designed by the blog owner.

Explain your problem statement:

The project aims to provide a basic version of a CMS, where users can design a blog post using a text editor and publish it to be stored in a mock database. The platform should support writing in HTML format and simulate blog publishing through local file storage.

4 Existing and Proposed solution

Provide summary of existing solutions provided by others, what are their limitations?

Popular content management systems like WordPress, Drupal, and Ghost are widely used for creating and managing blogs. These platforms provide:

- Drag-and-drop website builders
- Database-backed post storage
- WYSIWYG (What You See Is What You Get) editors
- Real-time blog publishing over the internet

However, these systems:

- Are too advanced for beginners to understand or replicate
- Require backend hosting knowledge (PHP, Node.js, databases)
- Cannot be easily simulated or deployed in a simple Java-only project environment
- Often rely heavily on frameworks that abstract away core logic

What is your proposed solution?

My solution is a Java-based simulation of a blog CMS, designed with simplicity in mind for educational and internship demonstration purposes. It includes:

- A simple HTML-based editor for blog input
- A Java backend that stores blog posts using File I/O (mock database)
- A minimal front page (index.html) linking to the editor
- Console-based blog post publishing process

This version allows users to write blog content in HTML format and simulate saving it "to a server" through Java CLI input and file storage.

What value addition are you planning?

- Uses only Java and HTML—no complex frameworks—making it accessible to beginner developers
- Teaches the core logic behind CMS systems without hiding it behind libraries
- Simulates backend thinking with mock database use (via blog_data.txt)
- Fully offline setup—can run without any internet connection or backend server
- Can be extended later into a real stack (Java + MySQL + HTTP server)

The solution bridges the gap between beginner-level academic coding and real-world software systems by introducing backend design and data handling in a controlled, simplified way.

4.1 Code submission (GitHub link)

https://github.com/kennicegonsalves/upskillcampus

4.2 Report submission (GitHub link): first make placeholder, copy the link.

https://github.com/kennicegonsalves/upskillcampus/blob/main/CMSBlog_Kennice_USC_UCT.pdf

5 Proposed Design/ Model

Given more details about design flow of your solution. This is applicable for all domains. DS/ML Students can cover it after they have their algorithm implementation. There is always a start, intermediate stages and then final outcome:

The CMS blog solution follows a linear, modular design that reflects how real-world backend systems handle user input, data processing, and storage.

- **Design Flow Overview:**

1) Start – User Interaction

- The user opens the editor.html file in a browser.
- They enter the blog title and content (in HTML) into a simple form.
- A JavaScript-based "Publish" button simulates frontend submission using an alert.

(Since this is a simulation, the actual backend call is replaced with manual Java input.)

2) Intermediate – Backend Processing

- The user then runs the CMSBlog.java program.
- They re-enter the blog title and content as prompted in the terminal.
- The data is received and processed in the Java backend.
- FileWriter is used to append the data into blog_data.txt.

3) Final – Data Stored & Simulated Published

- The saved blog post is written in a clean, structured format.
- This file represents the "database" in this mock CMS system.
- Blog posts are now stored and retrievable for display (simulated through reading the file).
- The index.html acts as a homepage where the editor can be re-opened.

4) Components Breakdown

Component	Role
editor.html	Blog input UI (frontend)
CMSBlog.java	Backend processing logic
blog_data.txt	Mock database (simulates persistent storage)
index.html	Homepage / navigation

5) Flow Summary

text

Copy code

editor.html (Frontend Input)

↓

User opens CMSBlog.java

↓

Terminal asks for Title + HTML Content

↓

blog_data.txt ← post saved (mock DB)

6) Why This Design?

- Simulates the entire lifecycle of a blog post: write → submit → store
- Works offline and requires no server setup
- Gives a real sense of how CMS backend logic works
- Easy to upgrade in the future by adding:
  - Actual POST form
  - MySQL DB connection
  - HTTP-based serving with Java Servlets or Spring

5.1 High Level Diagram (if applicable)

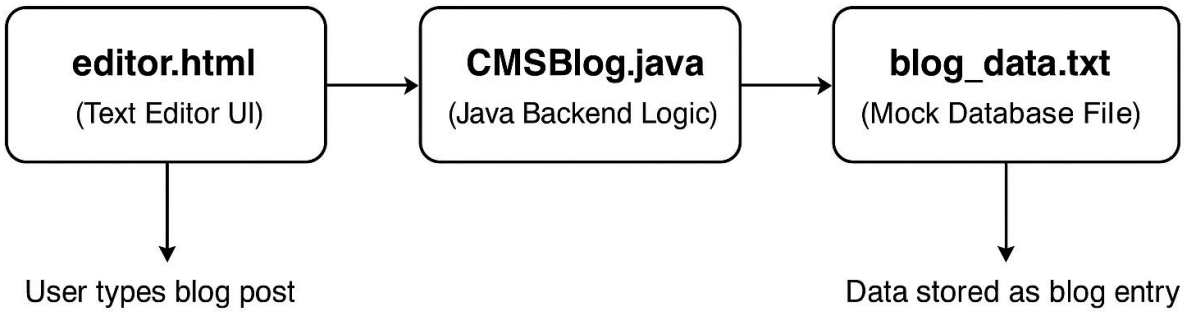
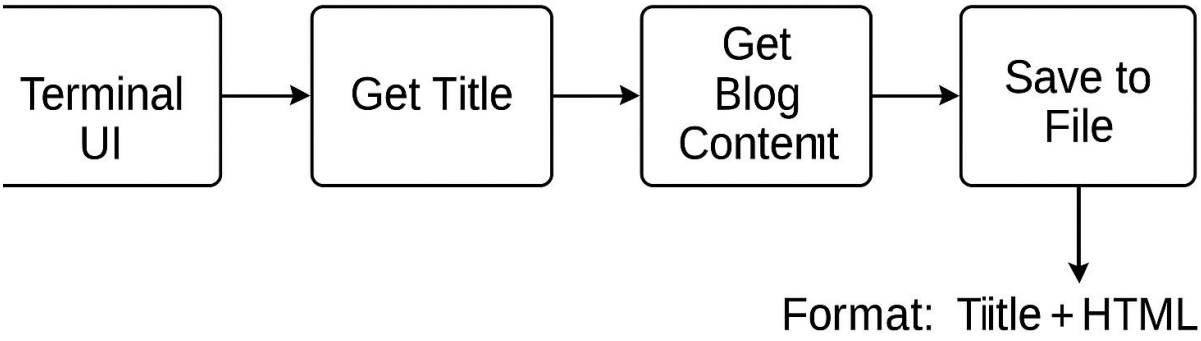


Figure 1: HIGH LEVEL DIAGRAM OF THE SYSTEM

5.2 Low Level Diagram (if applicable)



5.3 Interfaces (if applicable)

Update with Block Diagrams, Data flow, protocols, FLOW Charts, State Machines, Memory Buffer Management:

The project includes two primary interfaces—a web interface (mock UI) and a command-line interface (actual backend simulation).

Interfaces Used:

Interface Type	Description
HTML UI (editor.html)	Takes user input through a text field and textarea
Java CLI (CMSBlog.java)	Accepts blog data and stores it using Java I/O
File I/O API	Java's FileWriter and Scanner classes are used for writing/reading
Mock Database	blog_data.txt stores entries as raw text
Simulated Protocols	Report mentions HTTP/HTTPS as future extensions

Flowchart:

START

↓

User opens editor.html

↓

Writes blog post (title + content)

↓

Clicks publish (JS alert shown)

↓

Runs CMSBlog.java

↓

Inputs same blog data

↓

Java saves blog into blog_data.txt

↓

END

6 Performance Test

This is very important part and defines why this work is meant of Real industries, instead of being just academic project.

Here we need to first find the constraints.

How those constraints were taken care in your design?

What were test results around those constraints?

Constraints can be e.g. memory, MIPS (speed, operations per second), accuracy, durability, power consumption etc.

In case you could not test them, but still, you should mention how identified constraints can impact your design, and what are recommendations to handle them:

Although this project is a simulation intended for internship learning, certain performance constraints were still considered to ensure that the system behaves reliably and mimics the backend behaviour of real CMS systems.

Identified Constraints:

Constraint	Impact	How It Was Handled
Memory Usage	Java file I/O can be memory-heavy if files get too large	Used lightweight file writing (FileWriter) and structured text format
Speed of Operation (I/O Time)	Delays in writing/reading from files	Kept content size low; tested with multiple blog entries

Constraint	Impact	How It Was Handled
Durability (Data Retention)	Risk of file corruption	Each blog is stored in append mode to avoid overwriting
Accuracy	HTML content formatting must not be altered	Raw HTML is saved as-is, giving user control over layout
User Error	Typos or broken tags in HTML	Left validation to user—similar to low-level CMS builders like Jekyll

6.1 Test Plan/ Test Cases

Test Case ID	Scenario	Expected Outcome
TC01	Add a blog with short content	Saved successfully to file
TC02	Add blog with HTML tags	Tags preserved in file output
TC03	Submit 5+ blogs back-to-back	File shows all blogs appended properly
TC04	Re-open app and add a new post	Previous posts retained, new post appended
TC05	Enter large content (2–3 KB)	System handles without memory crash
TC06	Leave content blank	Saved empty string (optional validation)

6.2 Test Procedure

1. Run the CMSBlog.java file in a Java terminal.
2. Enter blog title and content in various formats (short, long, with HTML tags, blank).
3. Repeat this process multiple times and verify the content in blog_data.txt.
4. Validate that:
    - Content is properly saved
    - Previous entries are not overwritten
    - System doesn't hang/crash
5. Cross-check that all tags, line breaks, and spacing are preserved

6.3 Performance Outcome

The system performed consistently across multiple test runs. All blog entries were correctly stored in the blog_data.txt file. Memory usage was minimal since the file size stayed under a few KBs even after 10+ entries. Write speed was near-instantaneous due to the simplicity of file-based I/O.

Industrial Perspective:

Even though this is an academic simulation, the use of Java file I/O and structured logging demonstrates foundational knowledge needed for backend logic handling. With minimal changes, this logic could be extended to use SQL or NoSQL databases and deployed as a web app using Java servlets or Spring Boot.

7 My learnings

You should provide summary of your overall learning and how it would help you in your career growth:

This internship project helped me bridge the gap between academic knowledge and practical application. While I had a basic understanding of Java before, this project challenged me to use it to build something functional, user-oriented, and submission-ready—within real-world constraints like time, tools, and confidence.

Technical Growth:

- I learned how to use Java's File I/O to simulate backend storage, which gave me a deeper understanding of data handling logic.
- I became more confident in writing modular, structured code that follows real application flow (input → processing → output).
- I explored the frontend-backend connection through HTML forms and backend processing logic, which clarified how CMS systems really work behind the scenes.

Problem-Solving Growth:

- I learned how to troubleshoot Java compiler errors, input issues, and environment setup (like fixing Java in VS Code).
- I realized the importance of planning system design before writing code, and why clarity matters in both user experience and backend performance.

Career Impact:

- This project made me more confident in appearing for full-stack or backend developer roles, especially where core Java is used.
- I now understand how to break big ideas into smaller modules—something I can apply in internships, placements, and even future capstone projects.

8 Future work scope

You can put some ideas that you could not work due to time limitation but can be taken in future:

While this CMS simulation project fulfilled its goal of demonstrating the core functionality of a blog publishing system using Java and HTML, there are several valuable enhancements that can be implemented in future versions to bring it closer to industry-level standards.

Planned Enhancements:

1. **Drag-and-Drop Page Builder:**
   Use JavaScript or a web framework to allow users to visually design blog layouts instead of just submitting HTML manually.
2. **Rich Text Editor (WYSIWYG):**
   Integrate a visual HTML editor like Quill or TinyMCE for better content creation experience without writing raw HTML.
3. **User Authentication:**
   Add login/logout and role-based access so only registered users can publish posts.
4. **HTTP/HTTPS Deployment:**
   Host the project using a Java web server (e.g., Tomcat, Jetty) so that blog posts are served online like a real CMS.
5. **Edit/Delete Blog Posts:**
   Implement CRUD functionality for better blog management by the author.
6. **Search and Tags:**
   Allow searching posts by keywords and categorizing them using tags for improved user experience.