A REPORT OF ONE MONTH TRAINING
at
Ryaz.io Technologies

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENT
FOR THE AWARD OF THE DEGREE OF

BACHELOR OF TECHNOLOGY

Computer Science and Engineering

JUNE-JULY , 2025

Submitted by : Ravjot Singh

CRN - 2315191

URN - 2302646

DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING

GURU NANAK DEV ENGINEERING COLLEGE , LUDHIANA

(An Autonomous College Under UGC ACT)

CERTIFICATE

Declaration

I hereby certify that the work which is being presented in this training report for partial fulfillment of requirements for the award of degree of B.Tech. (Computer Science) submitted to the Department of Computer Science Engineering at **GURU NANAK DEV ENGINEERING COLLEGE, LUDHIANA under I.K. GUJRAL PUNJAB TECHNICAL UNIVERSITY** is an authentic record of my own work at Ryaz.io Technologies , Ludhiana, during 4-Week Industrial (Summer) Training - TR-102.

Ravjot Singh

URN-2302646

CRN-2315191

ABSTRACT

This report documents the work carried out during the one-month industrial training at Ryaz.io Technologies, focusing on backend development. Core web-based projects were developed using Node.js, Express, MongoDB, Socket.IO, and CRDTs. These projects included a URL Shortener,, Messaging App, and Collaborative Text Editor. The report highlights the daily progress, key technologies learned, challenges faced, and the outcomes achieved. Emphasis was placed on modular design, secure authentication, real-time communication, and collaborative synchronization. This training helped improve backend development skills and deepened understanding of real-world web application workflows.

Key Learnings :

1. **Backend Development with Node.js and Express**
   - Developed RESTful APIs with route handling, middleware, and modular controller structure.
   - Gained experience in server setup, routing strategies, and request/response lifecycle management.
2. **Authentication and Authorization**
   - Implemented secure user authentication using **JWT**, **bcrypt**, and **cookie-based sessions**.
   - Learned role-based access control and session persistence strategies.
3. **Database Design and MongoDB**
   - Designed schemas using **Mongoose**, used aggregation pipelines, and optimized queries.
   - Understood relational references (e.g., blog-user, messages-sender) and embedded vs referenced document trade-offs.
4. **Real-time Communication**
   - Built a **real-time messaging system** using **Socket.IO**.
   - Implemented public and private chats, message timestamping, and dynamic room handling.
5. **Collaborative Editing with CRDTs**

- Learned about **Conflict-free Replicated Data Types (CRDTs)** for real-time collaboration.
- Integrated **Yjs** for syncing document updates across multiple users using WebSockets.

6. **Frontend Integration**
   - Designed minimal frontends for testing real-time features and data flow between backend and client.
7. **File Handling and Cloud Storage**
   - Used **multer** for handling file uploads and **Cloudinary** for external file storage.
   - Dealt with private/public asset permissions and secure download/preview logic.
8. **Debugging and Problem Solving**
   - Learned to trace errors through console.log(), network tools, and GitHub Copilot suggestions.
   - Understood importance of incremental development, refactoring, and test-first approaches.
9. **Modular Code Architecture**
   - Practiced structuring apps into controllers, routes, services, and models.
   - Ensured maintainability, scalability, and ease of debugging across all projects.
10. **Version Control and Collaboration Tools**

- Used **GitHub** for code management and referencing.
- Understood how to use online developer resources effectively (e.g., GitHub issues, StackOverflow).

Acknowledgement

I take this opportunity to express my sincere gratitude to all those who have supported me throughout the course of my one-month industrial training.

First and foremost, I am deeply thankful to **Ms. Uttamdeep Kaur**, my industry mentor at **Ryaz.io Technologies, Ludhiana**, for her constant guidance, valuable insights, and support during the training. Her mentorship was instrumental in enhancing my technical skills and understanding of real-world software development practices.

I would also like to extend my heartfelt thanks to the **Department of Computer Science and Engineering, Guru Nanak Dev Engineering College, Ludhiana**, for providing me with the opportunity to undergo this training and apply my academic knowledge in a practical environment.

I am especially grateful to the entire team at Ryaz.io Technologies for their welcoming environment, constructive feedback, and collaborative spirit. Working alongside professionals in a dynamic setting has been an enriching experience.

Lastly, I would like to thank my family and friends for their continuous motivation and encouragement throughout this journey.

This training has not only helped me strengthen my programming and problem-solving abilities but also provided exposure to the development process, team collaboration, and industry-standard tools and practices.

Ravjot Singh

CRN-2315191

URN-2302646

About Company

Ryaz.io Technologies is a reputed technology-driven company based in **Ludhiana, Punjab**, specializing in software development, web technologies, and IT training services. It is an online-based software development and training company that operates remotely, providing project-based learning and mentorship through virtual platforms without a physical office setup. The organization offers a range of services that include web application development, backend system architecture, API integration, cloud-based solutions, and full-stack training programs for students and professionals.

Founded with the vision to bridge the gap between academic knowledge and industrial requirements, it focuses on practical learning experiences, hands-on projects, and mentoring to equip students with industry-relevant skills. The company actively works with emerging technologies such as **React , next.js , node.js , fast Api , postgressSOL , mongoDB etc.**

Key highlights of the company include:

- Offering professional training in **MERN Stack**, **DevOps**, **Cloud Integration**, and **Real-Time Web Applications**.
- Hosting regular industrial training and internship programs for engineering and computer science students.
- Mentoring project-based learning to help trainees build complete end-to-end applications.
- Maintaining a team of experienced software developers, mentors, and industry professionals.
- Encouraging open-source contribution, agile practices, and peer-reviewed code.

The organization promotes a collaborative and innovative environment where learners are encouraged to ask questions, explore different solutions, and grow both technically and professionally.

Contents

CHAPTER 1: INTRODUCTION

1.1 Background

The rapid growth of digital technologies has transformed the way people interact, communicate, and access information. Web-based platforms, collaborative tools, and real-time applications have become central to various domains—from education and healthcare to business and entertainment. As a student pursuing a degree in computer science and engineering, gaining hands-on experience in these areas is crucial to bridge the gap between academic concepts and real-world application.

The one-month industrial training at Ryaz.io Technologies, Ludhiana was focused on full-stack web development using modern technologies such as Node.js, Express.js, MongoDB and Socket.IO. This training provided a practical platform to develop scalable and real-time applications.

1.2 Objectives

The major objectives of the industrial training were:

- To gain hands-on experience in backend development using JavaScript-based technologies.
- To understand the principles of RESTful API design and implementation.
- To develop real-time features using WebSockets for instant communication.
- To learn secure authentication practices using JWT and session tokens.
- To debug, modularize, and deploy code effectively.
- To enhance problem-solving skills through project-based learning and code refactoring.

1.3 Scope of Training

The scope of the training encompassed:

- Designing and implementing backend logic using Node.js and Express.
- Creating and managing databases using MongoDB and Mongoose ORM.
- Integrating authentication and session management for secure access.
- Handling file uploads and external cloud storage with Multer and Cloudinary.

- Developing and testing real-time communication using Socket.IO.
- Creating minimal front-end templates using EJS and integrating with backend APIs.
- Hosting, testing, and debugging the applications locally.

1.4 Tools and Technologies Used

Category	Tools/Technologies
Backend Framework	Node.js, Express.js
Database	MongoDB, Mongoose
Authentication	JSON Web Tokens (JWT), bcrypt, cookie-session
Real-time Communication	Socket.IO, WebSocket
File Upload/Storage	Multer, Cloudinary
Version Control	Git, GitHub
Development Tools	VS Code, Postman, Nodemon, Chrome DevTools

1.5 Relevance to Academic Curriculum

This training aligns closely with the core subjects of Information Technology such as:

- Web Technologies
- Data Structures and Algorithms
- Database Management Systems
- Software Engineering

By practically applying concepts from these subjects, the training helped enhance the learning outcomes of my academic curriculum and also prepared me for real-world industry challenges.

CHAPTER 2 – TRAINING WORK UNDERTAKEN

During my one-month industrial training at Ryaz.io Technologies, I explored and implemented multiple core aspects of backend web development. The training was structured around solving real-world problems using modern development stacks. The entire experience was divided across several focused themes, allowing me to develop deep insights into how various components of a full-stack application operate together.

2.1 Backend Application Architecture

- Designed and implemented multiple backend services using Node.js and Express.js.
- Established modular file structures to manage routes, controllers, middlewares, and models.
- Followed RESTful API conventions for designing endpoints for CRUD operations.
- Utilized Mongoose to model and interact with MongoDB collections efficiently.
- Integrated dotenv to manage environment variables securely.
- Implemented server-side rendering using EJS for dynamic web pages in some modules.

2.2 Database Design and Management

- Created multiple MongoDB schemas using Mongoose for entities like URLs, blogs, messages, users, and documents.
- Handled schema validations, default values, timestamps, and relationships across collections.
- Implemented advanced MongoDB operations such as aggregation pipelines, pagination, and filtering.
- Stored session data and user information securely in MongoDB to enable persistent login.

2.3 Authentication and Authorization

- Implemented user registration and login flows with proper validation.
- Secured APIs using JWT (JSON Web Tokens) for stateless authentication.
- Stored tokens in cookies and created middleware to verify tokens on protected routes.
- Used bcrypt to securely hash and store user passwords.
- Controlled access to routes and features based on user roles and session states (e.g., blog authorship, message ownership).

2.4 Real-Time Communication and WebSockets

- Built real-time messaging and collaboration features using Socket.IO.
- Implemented public chat functionality and extended it to support Direct Messaging (DM) using private rooms.
- Managed socket events like connection, disconnection, message broadcast, and room joining.
- Applied event-based architecture to decouple front-end and back-end interactions.
- Debugged challenges in event synchronization and route isolation for DMs.

2.5 Session and State Management

- Used express-session and connect-mongo to manage server-side sessions.
- Persisted sessions across reloads and interactions using secure cookies.
- Debugged issues related to session inconsistency across views, especially for rendering dynamic content.

2.6 File Upload and Cloud Integration

- Integrated Multer to support file uploads in chat and blog features.
- Configured Cloudinary for cloud storage of media files.
- Ensured file uploads were validated, saved, and previewed using appropriate MIME types.
- Resolved bugs related to duplicate file deletions and incorrect upload paths.
- Implemented download and preview functionality for stored media using public Cloudinary URLs.

2.7 Error Handling and Debugging

- Developed consistent error-handling middleware across all Express apps.
- Utilized tools like console tracing, Postman, and GitHub Copilot to detect logic errors.
- Learned to handle client/server communication failures, including file not found, session timeouts, and token expiration.
- Refactored and debugged controller logic to resolve authentication bugs and update failures.

2.8 Frontend Integration

- Built basic interfaces using HTML, CSS, and JS templates to test backend APIs.
- Rendered data conditionally using values passed from backend routes (e.g., user sessions, messages).
- Integrated features like dynamic blog pages, search and filtering, dashboard lists, and real-time message rendering.
- Connected backend logic to user interaction components such as buttons, input fields, and status indicators.

2.9 Deployment and Testing

- Tested all applications locally with Nodemon, Postman, and socket clients.
- Refactored codebases into logical modules for reusability and deployment readiness.
- Emulated live user behavior through test UIs and simultaneous client connections.
- Deployed a few components to local production setups to simulate live environments.

2.10 URL Shortener Apis

The URL shortener project aimed to reduce long URLs into compact, shareable links and handle redirection to the original addresses.

Key Tasks and Learnings:

- Set up a backend using Node.js, Express, and MongoDB.
- Utilized the nanoid library to generate unique short codes for URLs.
- Designed a schema to store the original URL, its short version, and metadata like creation time and click count.
- Explored the Node.js crypto module to build a custom URL shortener function without relying on third-party libraries.
- Implemented redirect logic to route users from short URLs to original links.
- Added an analytics tracker to count the number of times a short URL was accessed.
- Faced and resolved file import issues caused by naming conflicts with built-in modules (url.js renamed to URL.js).
- Created a simple frontend for URL input and short link generation.
- Refactored the code into a modular design for better maintainability.

2.11 Real-Time Messaging Application

This application was designed to facilitate real-time public and private communication between users.

Key Tasks and Learnings:

- Built a Socket.IO-powered chat system with message broadcasting, storage, and retrieval.
- Designed a message schema containing sender name, content, and timestamp.
- Added support for user sessions using express-session and MongoDB, enabling login persistence.
- Developed functionality to display the sender's name and time for each message on the frontend.
- Implemented edit and delete features, accessible only to the original sender through session validation.
- Added file upload functionality using multer.
- Resolved encoding issues where uploaded files returned base64 instead of preview links.
- Introduced Direct Messaging (DMs) using private socket rooms to allow private communication.
- Reorganized routing logic for better readability and maintenance.
- Created a testing interface to simulate user chats, validate DMs, and track message delivery in real time.

CHAPTER 3: RESULTS AND DISCUSSION

3.1 Functional Applications Developed

- **URL Shortener Service:**
  A fully functional backend system that accepts long URLs and returns shortened versions. It supports custom code generation and redirection. The final system includes click tracking and analytics, along with a simple frontend form.

- **Messaging Application (Public & Private Chat):**
  A real-time chat app developed using Socket.IO, with support for public channels, direct messaging (DM), session-based message access control, and file uploads via Cloudinary. Users could send, edit, and delete messages with proper permission control.

3.2 Skills and Technologies Applied

- **Backend-Development:**
  Gained expertise in Node.js and Express.js by building scalable APIs, organizing code into modular components, and integrating middleware for validation and error handling.

- **Database-Management:**
  Designed and implemented MongoDB schemas using Mongoose. Understood data modeling, relationship handling, and advanced queries like aggregation and pagination.

- **Authentication.and.Security:**
  Learned and implemented secure authentication using JWTs and password hashing with bcrypt. Also handled token verification in real-time systems (sockets), which was a crucial learning experience.

- **Real-Time Communication:**
  Built real-time application using WebSockets and Socket.IO. Understood event-driven programming and how client-server synchronization works in practice.

- **File Uploads and Cloud Integration:**
  Integrated Multer for file handling and Cloudinary for storage and preview.

3.3 Key Challenges Faced

- **Session and State Handling:**
  Managing session data between backend and frontend, especially when rendering

dynamic content or updating socket connections, led to multiple debugging cycles.

- **Asynchronous Bugs and Token Handling:**
  Several issues occurred due to async logic errors, token expiration, and failure to attach auth headers to WebSocket clients. These were resolved using log tracing and tool-based diagnosis.

- **File Handling Errors:**
  In the messaging app, the uploaded files initially returned base64 strings or failed to preview due to incorrect pathing and configuration.

- **Frontend and Backend Sync:**
  Problems arose when the frontend logic didn't match backend expectations—particularly in socket rooms and dashboard rendering. These were fixed by revisiting API contracts and reviewing state flow.

3.4 Overall Outcome

By the end of the training period, I successfully delivered backend-driven application with production-ready features. The experience helped solidify my understanding of web development fundamentals, real-time communication, authentication systems, and collaborative technologies. Beyond technical skills, I learned how to debug effectively, modularize large codebases, and document my work systematically.

CHAPTER 4: CONCLUSION AND FUTURE SCOPE

4.1 Conclusion

Over this one-month training , I developed a deeper understanding of backend development, real-time communication, and collaborative systems. By completing four core projects, I explored multiple layers of application architecture including RESTful APIs, session handling and file storage. The experience helped me strengthen not only my technical proficiency but also my debugging and problem-solving capabilities.

4.2 Future Scope

- Add roles and dashboards to blogs and chat apps.
- Deploy apps using Docker and CI/CD pipelines.
- Enhance editor with rich text formatting and document history.
- Implement E2E encryption for messaging.
- Explore P2P sync and offline storage for collaborative tasks.