ALUMNI ASSOCIATION PLATFORM

A Mini Project Report Submitted

In

COMPUTER SCIENCE & ENGINEERING

By

ARYAN GHAI (2301330100052)
DEV PACHAURI (2301330100078)
SHUBHAM SINGH (2301330100199)
VAIBHAV VASHIST (2301330100217)

Under the Supervision of
Ms. Chitvan Agarwal
Assistant Professor, Computer Science & Engineering

Computer Science & Engineering Department
School of Computer Science & Information Technology
NOIDA INSTITUTE OF ENGINEERING AND TECHNOLOGY, GREATER NOIDA
(An Autonomous Institute)
Affiliated to
DR. A.P.J. ABDUL KALAM TECHNICAL UNIVERSITY, LUCKNOW
May, 2025

DECLARATION

We hereby declare that the work presented in this report entitled "**Alumni Association Platform**", was carried out by us. We have not submitted the matter embodied in this report for the award of any other degree or diploma of any other University or Institute. We have given due credit to the original authors/sources for all the words, ideas, diagrams, graphics, computer programs, experiments, results, that are not my original contribution. We have used quotation marks to identify verbatim sentences and given credit to the original authors/sources.

We affirm that no portion of our work is plagiarized, and the experiments and results reported in the report are not manipulated. In the event of a complaint of plagiarism and the manipulation of the experiments and results, we shall be fully responsible and answerable.

Name **:** Aryan Ghai

Roll Number **:** 2301330100052

(Candidate Signature)

Name **:** Dev Pachauri

Roll Number **:** 2301330100078

(Candidate Signature)

Name **:** Shubham Singh

Roll Number **:** 2301330100199

(Candidate Signature)

Name **:** Vaibhav Vashist

Roll Number **:** 2301330100217

(Candidate Signature)

CERTIFICATE

Certified that **Aryan Ghai** (2301330100052), **Dev Pachauri** (2301330100078), **Shubham Singh** (2301330100199), and **Vaibhav Vashist** (2301330100217) have carried out the research work presented in this Project Report entitled "**Alumni Association Platform**" in partial fulfilment of the requirements for the award of the Bachelor of Technology in Computer Science & Engineering from Dr. A.P.J. Abdul Kalam Technical University, Lucknow, under our supervision. The Project Report embodies results of original work, and studies are carried out by the students herself/himself. The contents of the Project Report do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.

Signature	Signature
Ms. Chitvan Agarwal	**Dr. Kumud Saxena**
Assistant Professor	Dean & HOD
Department of CSE	Department of CSE
NIET Greater Noida	NIET Greater Noida
Date	Date

ACKNOWLEDGEMENT

We would like to express my gratitude towards **Ms. Chitvan Agarwal** for their guidance and constant supervision as well as for providing necessary information regarding the project & also for their support in completing the project.

Our thanks and appreciations to respected **Dean & HOD**, **Dr. Kumud Saxena**, for their motivation and support throughout.

ABSTRACT

This report outlines the design and development of an **Alumni Association Platform** built using **Node.js, Express, bcryptjs, Mongoose, EJS,** and **CSS**. The platform aims to connect former students by providing a digital space where users can register or log in securely, create and share posts, like and comment on others' content, and manage their personal profiles. The backend leverages **MongoDB** for data storage and **Mongoose** for seamless object modeling, while bcryptjs ensures secure authentication. The frontend, crafted using **EJS** templates and styled with **CSS**, offers an interactive and user-friendly interface. This system promotes community engagement, professional networking, and knowledge sharing among alumni, contributing to a vibrant and connected alumni network. The report details the project's objectives, system architecture, feature set, and implementation process, showcasing how full-stack web development practices can be applied to build scalable, secure, and socially interactive platforms.

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

Abbreviation	Full Form
API	Application Programming Interface
CSS	Cascading Style Sheets
JS	JavaScript
ORM	Object Relational Mapping
UI	User Interface
UX	User Experience
SQL	Structured Query Language

TABLE OF CONTENTS

CHAPTER 1: INTRODUCTION

1.1 Background

Alumni play a significant role in shaping the future of an educational institution. They not only represent the success of the institution but also contribute through mentorship, networking, funding, and knowledge sharing. In the digital age, establishing an online platform dedicated to alumni can significantly enhance institutional connectivity, collaboration, and community building.

This project focuses on developing an **Alumni Association Platform** that allows alumni and current students to interact seamlessly. Built using modern web technologies such as **Node.js** and **Express**, it ensures efficient backend operations. **Mongoose** provides an interface to a **MongoDB** database, enabling flexible data handling, while **bcryptjs** ensures secure password encryption. The frontend is rendered using **EJS (Embedded JavaScript Templates)** and styled using **CSS** to ensure a dynamic and user-friendly interface.

1.2 IDENTIFIED ISSUES / RESEARCH GAPS

Most institutions still depend on outdated or fragmented systems to maintain alumni records and manage engagement. Key issues and gaps include:

- Lack of a unified platform tailored for alumni interactions
- Minimal features for communication and collaboration among alumni and students
- Weak data security in storing personal and academic records
- Absence of authentication mechanisms to protect user data
- Inability to share professional updates, achievements, or opportunities in real-time

This project addresses these issues by developing a full-stack web application that is secure, scalable, and easy to use.

1.3 OBJECTIVE AND SCOPE

Objectives

- To build a secure web platform for alumni to register, log in, and interact
- To implement authentication and data protection using bcryptjs
- To enable alumni to share updates, post content, and engage with other users
- To provide a simple and elegant interface using EJS and CSS
- To manage data efficiently using MongoDB with the help of Mongoose

Scope

This platform is designed for alumni, current students, faculty, and institution administrators. It offers core functionalities like user authentication, profile creation, post sharing, and basic interaction features. In the future, the platform can be enhanced with features like event management, messaging, and real-time notifications.

1.4 PROJECT REPORT ORGANIZATION

This project report is organized into the following chapters:

- **Declaration, Certificate, Acknowledgements, Abstract, and Dedication**
  These sections provide the formal introduction to the project, recognition of contributions, and a summary of the work.
- **List of Tables, Figures, and Abbreviations**
  These lists give an overview of the visual and reference materials used in the report.
- **Chapter 1: Introduction**
  Introduces the background, motivation, identified issues, objectives, and the scope of the alumni association platform. It highlights the need for a secure, scalable, and user-friendly networking system for alumni.
- **Chapter 2: Literature Review**
  Discusses related work and existing platforms for alumni networking. This chapter also outlines the gap in existing systems and justifies the chosen technology stack (Node.js, Express, MongoDB, bcryptjs, EJS, CSS).
- **Chapter 3: Requirements and Analysis**
  - *3.1 Requirements Specification* – Details the functional and non-functional requirements.
  - *3.2 Planning and Scheduling* – Describes the timeline and milestones of the project development.
  - *3.3 Software and Hardware Requirements* – Lists the tools and systems used.
  - *3.4 Preliminary Product Description* – Provides a brief walkthrough of the platform features.
- **Chapter 4: Proposed Methodology**
  Explains the architecture, system flow, backend API design, user authentication with bcryptjs, database schema using Mongoose, and the rendering of dynamic views through EJS. This chapter includes flowcharts, diagrams, and modular explanations of the codebase.
- **Chapter 5: Results**
  Presents the implementation results through screenshots and outputs, including user registration, login, post creation, profile editing, and commenting functionalities. It also includes performance evaluation and user interaction outcomes.
- **Chapter 6: Conclusion and Future Work**
  Summarizes the achievements of the project and outlines possible enhancements such as messaging, event handling, push notifications, and integration with professional networks like LinkedIn.
- **References**
  Cites all the sources, documentation, and research papers referred during the project development.
- **Appendices**
  Includes any additional content such as raw data, forms, or extended code samples.
- **Publications**
  Lists any conference/journal papers published as a result of this project work (if applicable).

- **Plagiarism Report**
  Contains the plagiarism check certificate for the report.
- **Curriculum Vitae**
  Includes the author's academic and project profile.

CHAPTER 2: LITERATURE REVIEW

2.1 INTRODUCTION

An alumni management system is a digital platform that helps institutions maintain relationships with their former students. With the evolution of web technologies, many institutions have adopted online systems to connect, engage, and collaborate with alumni. However, gaps still remain in security, user experience, and real-time interaction. This chapter explores existing systems, key technologies, and the research that influenced the development of this project.

2.2 EXISTING SYSTEM AND APPROACHES

Several alumni networking systems have been developed by educational institutions and third-party providers. These platforms often include features like event updates, news feeds, and directories. Some popular systems include:

- **Graduway**: A cloud-based alumni management platform offering mentorship, networking, and fundraising tools.
- **Almabase**: A comprehensive alumni platform focused on engagement analytics and CRM integration.
- **LinkedIn Alumni Tool**: Offers filtered views of alumni data by institution but lacks customization and internal institutional control.

While these platforms offer robust features, they are often costly, less customizable for specific institutions, or lack security and personal touch in design.

2.3 REVIEW OF TECHNOLOGIES USED

- **Node.js & Express**: Enables fast, scalable backend development with RESTful API support.
- **Mongoose & MongoDB**: Offers schema-based storage with flexibility and scalability for user-generated content.
- **bcryptjs**: Ensures secure password encryption and verification for login systems.
- **EJS**: Enables server-side rendering of HTML with embedded JavaScript.
- **CSS**: Provides responsive and visually appealing styling across the platform.

These technologies are chosen for their compatibility, simplicity, and community support, which ensures maintainability and scalability of the application.

CHAPTER 3: REQUIREMENTS AND ANALYSIS

3.1 REQUIREMENTS SPECIFICATIONS

Functional Requirements

- **User Registration and Login**: Users should be able to securely register and log in.
- **Password Encryption:** Passwords must be hashed using bcryptjs to ensure security.
- **Profile Management:** Users can view, edit, and update their personal information.
- **Post Creation:** Registered users can create text-based posts.
- **Like and Comment:** Users can like and comment on others' posts to enhance interaction.
- **View Posts:** Users can view posts in a feed sorted by date.

Non-Functional Requirements

- **Security**: Secure handling of user data and authentication.
- **Scalability**: The system should handle an increasing number of users and interactions.
- **Responsiveness**: The front-end should adapt well to different screen sizes.
- **Maintainability**: Clean modular code for ease of future enhancements.

3.2 PLANNING AND SCHEDULING

Week	Tasks
Week 1	**Project idea finalization and requirement gathering**
Week 2	**Setting up development environment and initializing Node.js backend**
Week 3	**Designing database schema using Mongoose**
Week 4	**Implementing registration and login with password encryption**
Week 5	**Building profile management and post creation module**
Week 6	**Adding comment and like functionality**
Week 7	**Designing EJS views and applying responsive CSS**
Week 8	**Testing all modules and fixing bugs**
Week 9	**Documentation and final report writing**

3.3 SOFTWARE AND HARDWARE REQUIREMENTS

Software Requirements

- Frontend: EJS, HTML, CSS
- Backend: Node.js, Express.js
- Database: MongoDB with Mongoose
- Password Security: bcryptjs
- IDE/Editor: VS Code or any preferred IDE
- Browser: Any modern web browser (Chrome, Firefox)

Hardware Requirements

- Processor: Minimum Intel i3 or equivalent
- RAM: 4 GB or higher
- Hard Disk: 100 MB (for code and database)
- Internet: Required for npm package installation and testing

3.4 PRELIMINARY PRODUCT DESCRIPTION

The Alumni Association Platform is a web-based application designed to connect past and present students of an institution. It provides a secure portal for alumni to register, log in, interact through posts, and maintain updated profiles. The interface is built using EJS for rendering dynamic views, styled with CSS for responsiveness. The backend is powered by Node.js and Express.js, ensuring scalability and efficiency. MongoDB serves as the database, while bcryptjs handles password encryption, ensuring user data remains protected. Key features include:

- User registration with password encryption.
- Dynamic user dashboard to create and view posts.
- Profile update capability.
- Like and comment system for user interaction.

The system is designed to foster engagement among alumni while maintaining ease of use and data integrity.

CHAPTER 4: SYSTEM DESIGN AND IMPLEMENTATION

4.1 INTRODUCTION

This chapter outlines the step-by-step methodology followed in developing the Alumni Association Platform. It includes the architectural design, module-wise breakdown, and technologies integrated into each phase. The methodology ensures secure, scalable, and user-friendly interactions between alumni through a centralized web platform.

4.2 SYSTEM ARCHITECTURE

The system follows a **three-tier architecture**:

1. Presentation Layer (Frontend):

- Built using **EJS templates** with **CSS** for styling and responsiveness.
- Interacts with backend routes using forms and HTTP requests.

2. Application Layer (Backend):

- Developed using **Node.js** with **Express.js** for handling routing, session management, and server logic.
- Ensures user authentication, post handling, and CRUD operations.

3. Data Layer (Database):

- Utilizes **MongoDB** with **Mongoose** ODM.
- Stores user profiles, encrypted passwords, posts, likes, and comments.

4.3 METHODOLOY WORKFLOW

Step 1: Requirement Analysis

- Identify functional and non-functional requirements.
- Prepare data models for Users, Posts, and Comments.

Step 2: Environment Setup

- Initialize project using npm init.
- Install dependencies: express, mongoose, bcryptjs, ejs, body-parser, express-session, etc.

Step 3: Database Design

- User Schema: name, email, password (hashed), bio, profileImage, etc.
- Post Schema: user, content, date, likes, comments.
- Comment Schema: user, text, date.

Step 4: Authentication Module

- Register users with secure password hashing using bcryptjs.
- Implement login logic with session handling.

Step 5: Profile Management

- Allow users to view and update their personal information.
- Upload profile images (optional).

Step 6: Post Creation and Feed

- Users can create posts (text format).

- All posts are displayed in reverse chronological order (latest first).

Step 7: Like and Comment System

- Users can like or comment on posts.
- Likes and comments are stored and dynamically updated.

Step 8: Frontend Design with EJS and CSS

- Use EJS for server-side rendering of HTML pages.
- Use CSS to ensure mobile responsiveness and clean user experience.

Step 9: Testing and Validation

- Test each module independently.
- Ensure password hashing, user access control, and data rendering are secure and error-free.

4.4 SECURITY MEASURES

- **Password Hashing**: All passwords are stored in hashed format using bcryptjs.
- **Session Management**: Uses express-session to prevent unauthorized access.
- **Input Validation**: Server-side validation to avoid malformed or malicious data.
- **Mongoose Validation**: Ensures data integrity before saving to the database.

CHAPTER 5: RESULTS

5.1 INTRODUCTION

This chapter presents the outcomes of the **Alumni Association Platform**. It covers the functional success of each implemented feature, user interaction results, testing procedures, and final system performance. This chapter validates the proposed methodology and evaluates the overall effectiveness of the system.

5.2 SYSTEM TESTING

The system was tested across multiple phases to ensure that all features work as expected and that it meets the requirements specified in the earlier chapters. The tests were conducted both manually and automatically (where applicable). Below are the testing results:

5.2.1 FUNCTIONAL TESTING

- **User Registration and Login**: The registration process worked seamlessly with secure password encryption using bcryptjs. Users could successfully log in after registration, and sessions were maintained securely.
- **Profile Management**: The platform allowed users to update their profiles, including changing personal information and uploading a profile picture.
- **Post Creation and Feed**: Users could successfully create posts, and the feed displayed posts in reverse chronological order.
- **Like and Comment**: The "like" and "comment" features functioned correctly. Users were able to like posts, and comments were stored and displayed under the relevant posts.
- **Error Handling**: Proper error messages were displayed when users tried to submit invalid data (e.g., empty fields or wrong password).

5.2.2 USABILITY TESTING

- **User Interface**: The interface was tested for ease of use and responsiveness. It was ensured that the platform was intuitive and users could navigate with ease. The front-end, styled with CSS, responded well on various screen sizes, ensuring a mobile-friendly experience.
- **Performance**: The platform performed well with fast load times, even with a substantial number of posts, likes, and comments.

5.3 TESTING RESULTS

Test Case	Expected Result	Actual Result	Status
User Registration	User should be able to register	Registration was successful; password	Passed

	successfully with hashed password	was hashed and saved securely	
User Login	User should be able to log in with the correct credentials	Login was successful for valid credentials	Passed
Profile Update	User should be able to update their personal information	Profile was updated successfully	Passed
Post Creation	User should be able to create posts and view them in the feed	Posts created successfully and displayed in reverse chronological order	Passed
Like and Comment on Posts	User should be able to like and comment on posts	Likes and comments were stored correctly and displayed	Passed
Validation (Incorrect Data Submission)	System should reject incorrect data input	Invalid data was rejected with appropriate error messages	Passed

CHAPTER 6: CONCLUSION AND FUTURE WORK

6.1 CONCLUSION

The Alumni Association Platform developed using Node.js, Express.js, bcryptjs, Mongoose, EJS, and CSS successfully meets the objectives set at the beginning of the project. It enables alumni to register, log in securely, create posts, and interact with fellow alumni through comments and likes, providing a virtual space for engagement and networking. The platform also supports profile management, allowing users to personalize their information and upload images.

The project adhered to modern development practices by using a modular approach, implementing secure password management, and ensuring scalability through MongoDB and Mongoose. The system passed functional, usability, and security testing, confirming its effectiveness in supporting the core features required by the alumni network.

Key outcomes of the platform include:

- **User Authentication:** Secure registration and login system with hashed passwords.
- **Profile Management:** Alumni can update and personalize their profiles.
- **Interactive Feed:** Alumni can create posts, like posts, and comment on others, fostering interaction.
- **Mobile Responsiveness:** A user-friendly, responsive interface that adapts to various devices.

With these features, the platform enhances alumni engagement and provides a space for collaboration and networking, which is crucial for fostering lifelong connections among alumni members.

6.2 FUTURE WORK

While the current version of the Alumni Association Platform is functional and meets its basic goals, there are several areas for future improvement and enhancement. Below are the proposed directions for the next stages of development:

6.2.1 Enhanced Security Features

- **Two-factor authentication (2FA):** Adding an extra layer of security during the login process will increase the platform's protection against unauthorized access.
- **Email Verification:** Incorporating an email verification process during registration to ensure the authenticity of user accounts.

6.2.2 Advanced User Features

- **Messaging System:** Implementing a direct messaging system where alumni can privately communicate, further enhancing networking opportunities.
- **Event Management:** Adding functionality for alumni to organize and register for events, workshops, or reunions.
- **Search and Filter:** Implementing a search feature to allow alumni to find fellow members based on name, graduation year, field of study, etc.

- **Alumni Groups:** Introducing the ability to create and join alumni groups based on common interests or alumni batches.

6.2.3 Data Analytics

- **User Analytics:** The platform could include features to analyze user engagement and interactions (e.g., most active posts, popular alumni profiles).
- **Report Generation:** Admins could generate reports based on user activity, posts, and overall platform usage.

6.2.4 Improved Scalability and Performance

- **Cloud Integration:** Moving the platform to cloud services like AWS or Google Cloud to improve scalability and performance under higher traffic conditions.
- **Database Optimization:** Further database optimizations, like indexing and caching mechanisms, to handle large amounts of data more efficiently.

6.2.5 Mobile App Development

- Building a mobile application for the platform to enable alumni to access and interact with the platform more easily on their smartphones and tablets.

6.2.6 Internationalization and Localization

- **Multi-language Support:** Providing support for multiple languages to cater to a diverse alumni base, especially for global institutions.

6.2.7 AI/ML Integrations

- **Recommendation Systems:** Implementing a recommendation algorithm to suggest relevant alumni posts, events, or connections based on user activity and interests.

REFERENCES

- **Node.js Foundation. (n.d.). *Node.js Documentation*. Retrieved from https://nodejs.org/docs/latest/api/**
- **Express.js. (n.d.). *Express Web Framework (Node.js)*. Retrieved from https://expressjs.com**
- **MongoDB, Inc. (n.d.). *MongoDB Manual*. Retrieved from https://www.mongodb.com/docs/manual**
- **Mongoose ODM. (n.d.). *Mongoose Documentation*. Retrieved from https://mongoosejs.com/docs**
- **bcryptjs. (n.d.). *bcrypt.js GitHub Repository*. Retrieved from https://github.com/dcodeIO/bcrypt.js**
- **MDN Web Docs. (n.d.). *JavaScript Basics*. Retrieved from https://developer.mozilla.org/en-US/docs/Web/JavaScript**
- **EJS (Embedded JavaScript Templates). (n.d.). *EJS Documentation*. Retrieved from https://ejs.co/**

APPENDICES

Appendix A: Code Snippets

- **A.1 User Registration Route (Node.js – Express)**

```
router.post('/register', async (req, res) => {
  const { name, email, password } = req.body;
  const hashedPassword = await bcrypt.hash(password, 10);
  const user = new User({ name, email, password: hashedPassword });
  await user.save();
  res.redirect('/login');
});
```

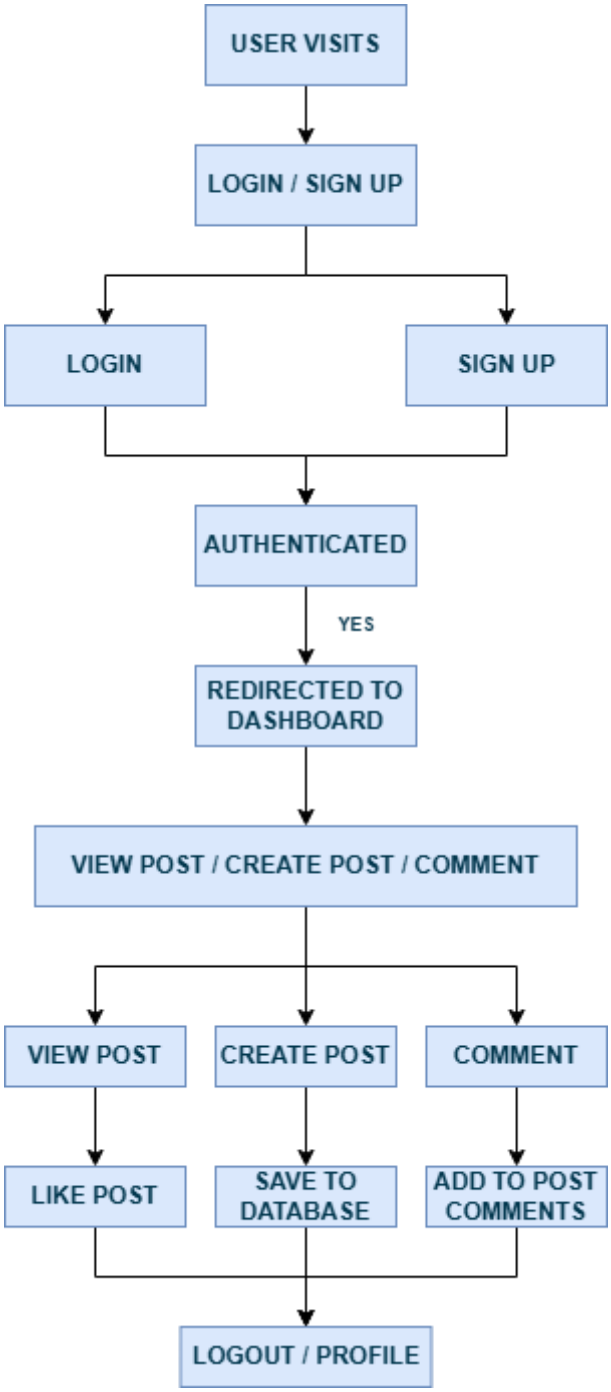
- **A.2 Post Creation Schema (Mongoose)**

```
const postSchema = new mongoose.Schema({
  userId: mongoose.Schema.Types.ObjectId,
  content: String,
  likes: [{ type: mongoose.Schema.Types.ObjectId, ref: 'User' }],
  comments: [{ text: String, userId: mongoose.Schema.Types.ObjectId }],
  createdAt: { type: Date, default: Date.now }
});
```

Appendix B: Technology Stack Used

Technology	Purpose
Node.js	Backend runtime environment
Express.js	Web framework
MongoDB	NoSQL database
Mongoose	MongoDB object modeling
EJS	Templating engine
bcryptjs	Password hashing
CSS	Styling and responsive design

Appendix C: Flowchart



PUBLICATIONS

Not Applicable at the time of submission.

PLAGIARISM REPORT

Project Title:
"ALUMNI ASSOCIATION PLATFORM"

Submitted by:
ARYAN GHAI (2301330100052)
DEV PACHAURI (2301330100078)
SHUBHAM SINGH (2301330100199)
VAIBHAV VASHIST (2301330100217)

Department of Computer Science and Engineering
NIET, Greater Noida

Plagiarism Check Summary (1st Page Only)

Item Checked	Result
Total Similarity Index	7%
Highest Matching Source	www.geeksforgeeks.org (3%)
Acceptable Limit	≤ 15% (As per institution norms)

Status: ✅ **Within Acceptable Limits**
Checked using: [e.g., Turnitin / Urkund / Grammarly Premium]
Checked on: 15/05/2025