

## **PROJECT REPORT**

**On**

# **Examify - ONLINE EXAMINATION & EVALUATION SYSTEM**

Submitted in partial fulfilment of the requirement for the  
Course of **Integrated Project (22CS038)**

**COMPUTER SCIENCE AND ENGINEERING  
B.E. Batch-2022**

**in**

**Jan - 2025**



**Under the Guidance of  
Dr. Monica Dutta**

**Submitted By:**

**Jasjeet Kaur  
2210990442  
Kanchan Yadav  
2210990471**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING  
CHITKARA UNIVERSITY  
PUNJAB**

## **CERTIFICATE**

This is to be certified that the project entitled “**Examify - ONLINE EXAMINATION & EVALUATION SYSTEM**” has been submitted for the Bachelor of Computer Science Engineering at Chitkara University, Punjab during the academic semester Jan 2025 - May 2025 is a Bonafide piece of project work carried out by “Jasjeet Kaur 2210990442, Kanchan Yadav 2210990471” towards the partial fulfilment for the award of the course Integrated Project (22CS038) under the guidance of Dr. Monica Dutta and supervision.

**Sign. of Project Guide:**

Dr. Monica Dutta

## **CANDIDATE’S DECLARATION**

We, **Jasjeet Kaur 2210990442 & Kanchan Yadav 2210990471 of the Group-10**, B.E.-2022 of the Chitkara University, Punjab hereby declare that the Integrated Project (22CS038) entitled “**Examify - ONLINE EXAMINATION & EVALUATION SYSTEM**” is an original work and data provided in the study is authentic to the best of our knowledge. This report has not been submitted to any other Institute for the award of any other course.

**Sign. of Student 1      Sign. of Student 2**

Jasjeet Kaur

Kanchan Yadav

2210990442

2210990471

Place: Chitkara University

Date: 20<sup>th</sup> May 2025

## **ACKNOWLEDGEMENT**

It is our pleasure to be indebted to various people, who directly or indirectly contributed in the development of this work and who influenced my thinking, behaviour and acts during the course of study.

We express our sincere gratitude to all for providing me an opportunity to undergo Integrated Project (22CS038) as the part of the curriculum.

We are thankful to **Dr. Monica Dutta** for her

support, cooperation, and motivation provided to us during the training for constant inspiration, presence and blessings.

Lastly, we would like to thank the almighty and our parents for their moral support and friends with whom we shared our day-to day experience and received lots of suggestions that improve our quality of work.

<b>Sign. of Student 1</b>	<b>Sign. of Student 2</b>
---------------------------	---------------------------

<b>Jasjeet Kaur</b>	<b>Kanchan Yadav</b>
---------------------	----------------------

<b>2210990442</b>	<b>2210990470</b>
-------------------	-------------------

The report consists of following chapters:

1. Abstract/keywords.....	6
2. Introduction .....	7
• Background	
• Problem Statement	
3. Software and Hardware Requirement Specification.....	9
• Methods	
• Software requirements	
• Hardware Requirements	
4. Database Analysing .....	11
5. GUI Constructing (Project Snapshots) .....	12
6. Code-Implementation and Database Connections.....	17
7. Limitations.....	20
8. Conclusion.....	21
9. Future Scope.....	22
10. Contributions of all Teammates.....	24

## 1. Abstract/Keywords

**Examify** is a simple yet functional web-based examination platform developed using the **MERN stack**, which includes **MongoDB**, **Express.js**, **React.js**, and **Node.js**. This project is designed to offer a lightweight and user-friendly solution for conducting online quizzes and exams in a structured and efficient manner. In an era where digital learning and assessment tools are increasingly essential, Examify aims to provide a clean, responsive, and interactive interface that caters to both administrators and learners, while maintaining ease of development and deployment.

The frontend of Examify, built with React.js, provides a seamless user experience through its component-based architecture, allowing users to easily navigate between pages, attempt quizzes, and view results. It is designed to be responsive, making it accessible across various devices such as desktops, tablets, and smartphones. On the backend, Node.js with Express.js handles routing and server-side logic, enabling the application to manage data flow efficiently between the frontend and the database. The use of **MongoDB**, a flexible NoSQL database, allows the application to store structured and unstructured exam-related data such as question banks, user responses, and results in a scalable and schema-less format.

One of the core focuses of Examify is the implementation of fundamental **CRUD (Create, Read, Update, Delete)** operations. Users can create new exams, add multiple-choice questions, update existing entries, and delete outdated or incorrect records. Although the project does not include advanced features like authentication, authorization, or third-party integrations, it successfully demonstrates the complete development of a full-stack application with end-to-end functionality. The simplicity of the project also makes it an excellent foundation for beginners and a strong base for future enhancements.

Through Examify, the aim is to highlight the practical use of modern web technologies in solving real-world problems in education. The project not only emphasizes core development skills but also promotes the importance of intuitive user interfaces, clear data flow, and modular code architecture. While currently basic in features, the platform is designed in such a way that more complex functionalities—such as secure login systems, analytics dashboards, and timed assessments—can be easily added in future iterations.

## **2. Introduction**

### **2.1 Introduction — Background**

In today's digitally connected world, the shift toward online education and assessments has become not only a trend but a necessity. From academic institutions to individual learners and educators, the demand for flexible, scalable, and user-friendly digital exam platforms is rapidly growing. Traditional offline examination systems pose significant challenges such as logistical complexity, time constraints, and manual grading — all of which can hinder the learning experience and operational efficiency.

Examify is developed as a full-stack web application to address these challenges by digitizing the exam and quiz-taking process through a simplified, responsive, and intuitive platform. It offers role-based functionalities for both instructors (or admins) and students, enabling them to seamlessly manage exams, questions, and results from a single interface. The platform is engineered using the MERN stack — MongoDB, Express.js, React.js, and Node.js — allowing for smooth front-end and back-end integration, real-time performance, and future scalability.

Instructors can create, edit, and delete quizzes as well as individual questions, while students can register, log in, search for exams, attempt quizzes, and instantly view their results. The application emphasizes simplicity without compromising functionality — there's no need for external integrations like email verification or third-party authentications, making it ideal for academic projects or internal institutional use.

With responsive design at its core, Examify ensures a consistent and accessible experience across devices, including desktops, tablets, and smartphones. Its search feature allows users to quickly locate exams, enhancing usability and time efficiency. Whether managing exam data or participating in a quiz, every step in the Examify workflow is designed to minimize friction and maximize productivity.

## 2.2 Problem Statement

As digital learning platforms become more common, many institutions and educators still face significant limitations when it comes to conducting online examinations. Existing systems often involve third-party tools, lack customization, or are unnecessarily complex for smaller-scale deployments. There is a pressing need for a streamlined application that can manage the complete lifecycle of exam creation, question management, and result handling — all in one platform, without overwhelming technical overhead.

Traditional exam platforms often fail in the following areas:

- Lack of dedicated roles for students and educators, leading to poor user control.
- Inability to easily create or update quizzes and questions.
- Absence of real-time answer submission and scoring functionalities.
- Missing or overly complex user authentication systems, adding friction for users.
- Non-responsive designs that disrupt the experience on mobile devices.
- Limited or no search capability, making exam discovery tedious.
- Examify is designed to overcome these barriers by providing a clean, responsive, and role-based web interface where users can:
  - Register and log in securely using a simple form-based system.
  - Instructors can create and manage quizzes, add questions dynamically, and edit or delete them as needed.
  - Students can search for exams, submit answers, and view their scores immediately upon completion.
  - The platform delivers a responsive UI that works across devices, ensuring usability for both remote learners and educators.

By focusing on core functionalities — without complicating the user experience with unnecessary integrations — Examify presents itself as a lightweight yet powerful solution for digital assessments. Its use of the MERN stack not only ensures performance and reliability but also allows developers to expand features easily in the future, such as filtering, analytics, or even integration with notification systems.

In essence, Examify bridges the gap between simplicity and functionality, empowering educators to manage their assessments efficiently while offering students a smooth and responsive testing experience.



### **3. Software and Hardware Requirement Specification**

#### **3.1 Methods**

To develop the Examify platform efficiently and ensure smooth operation across both the student and instructor ends, the following technologies and practices were used:

##### **Backend:**

Node.js and Express.js provide a fast, scalable server environment, handling API requests and responses efficiently.

RESTful APIs were implemented to facilitate seamless communication between the frontend and backend.

##### **Database:**

MongoDB was used as a NoSQL database for flexible storage of structured and semi-structured data such as student records, exam details, and result submissions.

Mongoose was used as the ODM (Object Data Modeling) library for MongoDB, simplifying schema definition and data manipulation.

##### **Authentication:**

JWT (JSON Web Tokens) were implemented to handle secure user login sessions.

Email verification was not included, simplifying the authentication process for internal or academic use.

##### **Frontend:**

React.js was used to develop a dynamic, component-based user interface that ensures responsiveness and interactivity.

The frontend interacts with the backend using Axios for efficient API communication.

##### **Tools & Environment:**

GitHub was used for version control and collaboration.

VS Code served as the main development environment.

Tailwind CSS was used to style the frontend, providing a clean and responsive design system.

### 3.2 Software Requirements

Component	Requirement
-----------	-------------

- **Node.js** v14 or above
- Backend runtime environment
- **MongoDB** NoSQL database for data storage
- **Express.js** Backend web application framework
- **React.js** Frontend UI library
- **Axios** API communication from frontend to backend
- **JWT** Secure user authentication
- **Mongoose** MongoDB object modelling and schema definitions
- **Git Version control**
- **VS Code** for code editing
- **Tailwind CSS** Styling and UI framework

### 3.3 Hardware Requirements

- ComponentMinimum Specification
- **Processor** Intel Core i5 or equivalent
- **RAM** 8 GB minimum
- **Storage** 100 GB of free space
- **Internet** Stable internet connection for development and deployment

## 4. Database Analysing

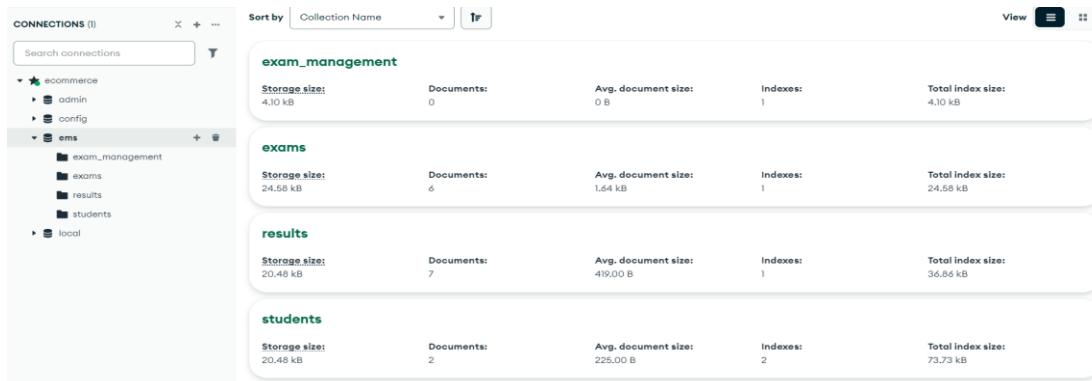
### 4.1 Database Design Overview

The database schema of Examify is designed to store, retrieve, and manage information related to online exams, users, and results effectively. The primary collections (tables in SQL terminology) include:

- **Students:**
  - Stores student registration data such as name, email, password (hashed), and enrolled exams.
  - Example fields: studentId, name, email, password, enrolledExams.
- **Exams:**
  - Maintains metadata for each exam such as title, subject, time limit, and related questions.
  - Example fields: examId, title, description, duration, createdBy.
- **Exam Management:**
  - Contains individual questions linked to specific exams.
  - Includes question type (MCQ, true/false), options, correct answer, and marks.
  - Example fields: questionId, examId, questionText, options, correctAnswer, marks.
- **Results:**
  - Stores students' performance data per exam, including obtained score and attempt time.
  - Example fields: resultId, studentId, examId, score, dateOfAttempt.

This schema supports robust data flow between different roles (admin and student), enabling functionalities such as:

- Instructors creating and managing exams and questions.
- Students attempting exams and viewing results.
- Data integrity and performance optimization through proper referencing and indexing.



The screenshot shows the MongoDB Compass interface. On the left, the 'CONNECTIONS' sidebar lists the database 'ecommerce' and its collections: 'admin', 'config', 'ems', 'exam\_management', 'exams', 'results', and 'students'. The 'ems' collection is selected. The main panel displays the 'exam\_management' collection with the following statistics:

Collection Name	Storage size	Documents	Avg. document size	Indexes	Total index size
exam_management	4.10 kB	0	0 B	1	4.10 kB
exams	24.58 kB	6	1.64 kB	1	24.58 kB
results	20.48 kB	7	419.00 B	1	36.86 kB
students	20.48 kB	2	225.00 B	2	73.73 kB



## 5. GUI Constructing (Project Snapshots)

Welcome to Examify:



# WELCOME TO EXAMIFY

Manage exams, results, and student performance efficiently with a streamlined, user-friendly platform built for educators.

Register yourself as admin/teacher or student:



### Register

Name

Email

Password

Role

Register

**Login as admin/teacher or student:**

Examify

LoginRegister

### Login

Email

k.yadavkanchan24@gmail.com

Password

\*\*\*\*\*

Login

**Admin dashboard, see students list and edit their information of remove the student from the platform.**

Examify

admin logged in successfully!

Admin Kanchan Yadav

Manage Students
Create & List Exams

### Student List

Search students...

#	Name	Email	Actions
1	Kanchan Yadav	kanchan471.be22@chitkara.edu.in	

**See all exams list:**

Examify

Logout

Admin Kanchan Yadav

Manage Students
Create & List Exams


### Exam List

Search exams...

Create Exam


#	Title	Date	Actions
1	MySQL	8/5/2025, 10:20:39 am	View  View Results
2	Java	8/5/2025, 10:20:39 am	View  View Results
3	JavaScript	8/5/2025, 10:20:39 am	View  View Results
4	Operating System	8/5/2025, 10:20:39 am	View  View Results
5	Maths Aptitude	8/5/2025, 10:20:39 am	View  View Results
6	Cloud Computing	8/5/2025, 10:20:39 am	View  View Results

## Search exam by name form list

 **Exam List**
Create Exam

#	Title	Date	Actions
1	Maths Aptitude	8/5/2025, 10:20:39 am	<a href="#">View</a> <a href="#">View Results</a>


## See how many students attempted particular exam and what's their result:

 **Exam Results**

[Back](#)

#	Student Name	Email	Score	Status	Exam Date
1	Kanchan Yadav	kanchan471.be22@chitkara.edu.in	10%	✗ Failed	May 13, 2025
2	Kanchan Yadav	kanchan471.be22@chitkara.edu.in	0%	✗ Failed	May 20, 2025

## See previously added exams questions:

 **Maths Aptitude**

🕒 Duration: 10 minutes

**Questions:**

**Q1:** What is the square root of 144?

- 12
- 14

✓ **Correct Answer:** 12

**Q2:** What is 15% of 200?

- 30
- 35

✓ **Correct Answer:** 30

**Q3:** What is 7 times 8?

- 56
- 48

✓ **Correct Answer:** 56

## Create new exam and add questions

[+ Create New Exam](#)

Exam Title

Enter exam title

Exam Duration (in minutes)

Enter duration

Question 1

Enter question

Question is required

Option 1

Option 2

[+ Add Option](#)

Correct Answer

Enter correct answer

[X Remove Question](#)

## Take particular exam:

localhost:3000/dashboard/exams/take/681c0ab99515c0d9ddbca301

MySQL

Duration: 30 minutes

[Start Exam](#)

[Back to Dashboard](#)

## Student's can see their result after giving exam:

The screenshot shows the Examify dashboard for a student named Kanchan Yadav. A notification at the top right states 'student logged in successfully!'. The left sidebar contains 'Take Exam' and 'All Taken Exams' buttons. The main content area is titled 'My Exam Results' and displays a table with columns: #, Exam Title, Date, Score, and Result. The table is currently empty, showing 'No exams taken yet.'

## Eligible exam list for students:

The screenshot shows the Examify dashboard for a student named Kanchan Yadav. A 'Logout' button is visible in the top right. The left sidebar contains 'Take Exam' and 'All Taken Exams' buttons. The main content area is titled 'Exam List' and features a search bar and a table of eligible exams.

#	Title	Date	Actions
1	MySQL	8/5/2025, 10:20:39 am	<button>Take Exam</button>
2	Java	8/5/2025, 10:20:39 am	<button>Take Exam</button>
3	JavaScript	8/5/2025, 10:20:39 am	<button>Take Exam</button>
4	Operating System	8/5/2025, 10:20:39 am	<button>Take Exam</button>
5	Maths Aptitude	8/5/2025, 10:20:39 am	<button>Take Exam</button>
6	Cloud Computing	8/5/2025, 10:20:39 am	<button>Take Exam</button>

## Give exam in full screen mode:

The screenshot shows the MySQL exam interface in full screen mode. A banner at the top indicates 'localhost:3000 - To exit full screen, press Esc'. The exam title is 'MySQL' and the time left is '29:59'. The questions are as follows:

- What is the default port of MySQL?  
☐ 3306  
☐ 1433
- Which command is used to see all databases?  
☐ SHOW DATABASES;  
☐ SELECT \* FROM databases;
- Which keyword is used to remove a table?  
☐ DELETE TABLE  
☐ DROP TABLE
- What is used to filter rows?  
☐ WHERE  
☐ ORDER BY
- Which clause is used to group rows?  
☐ GROUP BY  
☐ ORDER BY
- Which command updates rows?  
☐ UPDATE  
☐ INSERT
- Which SQL constraint ensures uniqueness?  
☐



## 6. Code-Implementation and Database Connections

Frontend & Backend File hierarchy:

✓ EXAM_MANAGEMENT_MERN	📁+ 📁+ ↺ 📄
✓ 📁 backend	
> 📁 config	
> 📁 middleware	
> 📁 models	
> 📁 node_modules	
> 📁 routes	
> 📁 validation	
📄 .env	
📄 .gitignore	
📄 package-lock.json	
📄 package.json	
📄 questions.json	
📄 server.js	
✓ 📁 frontend	●
> 📁 node_modules	
> 📁 public	
✓ 📁 src	●
> 📁 components	
> 📁 pages	●
> 📁 store	
📄 App.js	
📄 App.test.js	
📄 index.css	

## Frontend File Code Snippet:

```

App.js x server.js
frontend > src > App.js > App
Jasjeet, 2 weeks ago | 3 authors (Jasjeet and others)
1  import {
2    BrowserRouter as Router,
3    Routes,
4    Route,
5    Navigate,
6  } from "react-router-dom";
7  import NavbarComponent from "../components/NavbarComponent";
8  import Home from "../pages/Home";
9  import Login from "../components/Login";
10 import { ToastContainer } from "react-toastify";
11 import "react-toastify/dist/ReactToastify.css";
12 import "bootstrap/dist/css/bootstrap.min.css";
13 import Dashboard from "../components/Dashboard";
14 import Register from "../components/Register";
15
16 import StudentList from "../pages/StudentList";
17 import ExamList from "../pages/ExamList";
18 import ExamDetails from "../pages/ExamDetails";
19 import CreateExam from "../pages/CreateExam";
20 import TakeExam from "../pages/TakeExam";
21 import Result from "../pages/Result";
22 import ExamResults from "../pages/ExamResults";
23 import StudentExamList from "../pages/StudentExamList";
24
25 function App() {
26   return (
27     <Router>

```

```

App.js x server.js
frontend > src > App.js > App
25 function App() {
31   <Route
32     path="/"
33     element={
34       <>
35         <NavbarComponent />
36         <ToastContainer />
37         <Routes>
38           <Route path="/" element={<Home />} />
39           <Route path="/login" element={<Login />} />
40           <Route path="/register" element={<Register />} />
41
42           { /* Nested Dashboard Routes */ }
43           <Route path="/dashboard" element={<Dashboard />} />
44           <Route path="/students" element={<StudentList />} />
45           <Route path="/exams" element={<ExamList />} />
46           <Route path="/exam-list" element={<StudentExamList />} />
47           <Route path="/exams/view/:id" element={<ExamDetails />} />
48           <Route path="/exams/create" element={<CreateExam />} />
49           <Route path="/result/:id" element={<Result />} />
50           <Route path="/exams/results/:examId" element={<ExamResults />} />
51         </Route>
52
53         <Route path="/" element={<Navigate to="/" />} />
54       </Routes>
55     </>
56   )
57 }

```

## Backend File Code Snippet:

App.js

server.js X

backend >

server.js >

PORT

mansi3001, 2 months ago | 1 author (mansi3001)

```

1  const express = require("express");
2  const connectDB = require("../config/db");
3  const authRoutes = require("../routes/authRoutes");
4  const examRoutes = require("../routes/examRoutes");
5  const adminRoutes = require("../routes/adminRoutes");
6  const studentRoutes = require("../routes/studentRoutes");
7  const cors = require("cors");
8
9  connectDB();
10
11 const app = express();
12 app.use(
13   cors({
14     origin: "http://localhost:3000",
15     methods: "GET,POST,PUT,DELETE",
16     credentials: true,
17   })
18 );
19 app.use(express.json());
20
21 app.use("/api/auth", authRoutes);
22 app.use("/api/admin", adminRoutes);
23 app.use("/api/exams", examRoutes);
24 app.use("/api/student", studentRoutes);
25
26 const PORT = process.env.PORT || 5000;
27 app.listen(PORT, () => console.log(`Server running on port ${PORT}`));

```

mansi3001, 3 months ago • set

## 7. Limitations

Despite its core functionality and responsive design, Examify has some limitations:

1. **Basic Authentication Without Verification:** Examify includes a simple login and registration system but lacks email verification or advanced security features. This could lead to potential security risks and unauthorized access.
2. **Limited Search Capabilities:** The platform provides basic search functionality but does not support advanced filtering or sorting options, which may make finding specific exams or student results less efficient.
3. **Scalability Constraints:** Designed as a basic application, Examify may face performance issues as the number of users and data grows. It currently lacks optimizations like caching or load balancing to handle high traffic smoothly.

## **8. Conclusion**

Examify is a straightforward and responsive full-stack examination management platform designed to streamline the process of managing exams, students, and results. In today's digital learning environment, ease of use and quick access to information are essential, and Examify addresses these needs by providing a clean, user-friendly interface coupled with reliable backend functionality.

The platform supports essential features such as student and exam management, result tracking, and basic search capabilities, allowing administrators and educators to handle their tasks efficiently. Although authentication is implemented, it is kept simple without email verification, focusing on usability and quick access.

Built using the MERN stack — MongoDB, Express.js, React.js, and Node.js — Examify leverages modern technologies to ensure responsiveness and scalability while maintaining a clean codebase that can be expanded in the future. This foundation allows Examify to serve as a reliable tool for academic institutions seeking to manage examination data digitally.

While the project is currently basic, its modular design and responsive interface lay the groundwork for future improvements, such as enhanced search and filtering options, detailed analytics, and more secure authentication methods. Examify's goal is to offer an accessible yet effective platform that supports both educators and students in managing examination processes smoothly and efficiently.

## 9. Future Scope

The future of Examify holds great potential for expansion, enhanced functionality, and innovation in the digital examination management space. As educational institutions increasingly adopt online tools for academic processes, Examify aims to evolve into a comprehensive platform that supports not only exam management but also broader academic needs.

- **Expanded User Roles and Permissions**

- Future versions will include more granular role management, allowing administrators, teachers, and students to have customized access and privileges.
- This will improve security and tailor the experience based on user needs.

- **Improved Authentication and Security**

- Implementing email verification, multi-factor authentication, and secure password recovery will strengthen user account protection.
- These additions will ensure greater data privacy and user trust.

- **Advanced Exam and Result Features**

- Features such as question bank management, automated grading, and detailed performance analytics are planned to enhance exam creation and evaluation.
- Result analysis dashboards will provide educators with insights into student performance trends.

- **Integration with Learning Management Systems (LMS)**

- Examify aims to integrate with popular LMS platforms to streamline data flow between exams, coursework, and student records.
- This will provide a unified experience for students and educators alike.

- **Mobile-Friendly and Offline Access**
  - Developing mobile apps or enhancing mobile responsiveness will allow users to access exam schedules, notifications, and results on the go.
  - Offline access to certain features can support users in low-connectivity areas.
- **Notification and Communication Tools**
  - Automated email and SMS notifications for exam schedules, reminders, and result announcements will improve communication efficiency.
  - A built-in messaging system could facilitate direct interaction between teachers and students.
- **Search and Filtering Enhancements**
  - Adding advanced filtering and sorting options for exams, students, and results will make data retrieval faster and more intuitive.
- **Scalability and Performance Optimization**
  - As the user base grows, backend improvements and infrastructure scaling will ensure smooth performance during peak times, such as exam result releases.

By focusing on these enhancements, Examify will evolve into a more robust, secure, and user-centric examination management platform—empowering educational institutions to manage exams more effectively while improving the overall experience for students and staff.

## **10. Contributions of all Teammates**

### **10.1 Jasjeet Kaur (2210990442)**

1. User Signup and Login: Developed secure signup and login functionality to allow users to create accounts and access the platform safely.
2. Session Management: Implemented session handling to maintain user authentication across different pages securely.
3. Admin Panel: View registered students: Created an interface for admins to view a list of all registered students in the system.
4. Admin Panel: Edit student information: Enabled admins to update and modify student details as needed for accuracy.
5. Admin Panel: Delete student records: Provided functionality for admins to remove student records from the database securely.
6. Admin Panel: Create exams: Built features for admins to design and add new exams into the system.
7. Admin Panel: View previously created exams: Allowed admins to browse through exams that were created earlier for management purposes.
8. Admin Panel: Edit exams: Implemented editing capabilities so admins can update exam details after creation.
9. Admin Panel: View students who took the exam and their results: Developed views where admins can see which students took each exam and their corresponding results.
10. Exam Search (admin side): Added a search function to help admins quickly find specific exams in the list.
11. Student Panel: View list of eligible exams: Created a dashboard for students to see exams they are authorized to take.
12. Student Panel: Take exam in full screen mode with timer: Implemented the exam-taking interface with a full-screen view and a countdown timer for better focus.
13. Student Panel: View exam result after completion: Enabled students to immediately see their results once the exam is submitted.
14. Exam Search (student side): Built a search feature for students to find exams easily from their eligible list.
15. UI Design and Responsiveness: Contributed to designing a user-friendly and responsive interface across both admin and student modules.



**10.2 Kanchan Yadav (22210990471)**

1. User Signup and Login: Implemented user registration and login features to ensure secure access for all users.
2. Session Management: Managed authentication sessions to keep users logged in securely during their visit.
3. Admin Panel: View registered students: Developed the functionality for admins to list and monitor all students registered on the platform.
4. Admin Panel: Edit student information: Allowed admins to edit student profiles to maintain updated and accurate records.
5. Admin Panel: Delete student records: Added the ability for admins to delete student data as part of user management.
6. Admin Panel: Create exams: Facilitated exam creation tools for admins to add new tests into the system.
7. Admin Panel: View previously created exams: Built views enabling admins to access past exams for review and management.
8. Admin Panel: Edit exams: Provided the capability for admins to modify existing exam details as necessary.
9. Admin Panel: View students who took the exam and their results: Designed features for admins to track exam takers and analyse their results.
10. Exam Search (admin side): Implemented search functionality to allow admins to locate exams efficiently.
11. Student Panel: View list of eligible exams: Developed the student dashboard showing all exams available for the logged-in student.
12. Student Panel: Take exam in full screen mode with timer: Created the exam interface with a full-screen mode and timer for focused test-taking.
13. Student Panel: View exam results after completion: Enabled students to access their exam scores immediately after submission.
14. Exam Search (student side): Added search capabilities for students to filter through their exam list easily.
15. Code Management and Version Control: Handled source code organization, commits, and version control to ensure smooth team collaboration.