2. Analysis, Design Methodology and Implementation Strategy

2.1. Analysis and Comparison of existing and proposed system

Comparison:

Aspect	**Existing System**	**Proposed System**
Supervision	Limited real-time supervision, relying on student integrity	Smooth and secure user experience with intuitive features
Security	Basic measures, vulnerable to cheating	Enhanced security with live monitoring, preventing cheating
Communication	Slow and asynchronous communication via emails/messages	Real-time communication through instant chat for quick assistance
Interview	Proctors have limited means to intervene during exams	Proctors can swiftly intervene upon detection of anomalies
User Experience	May lack user-friendly features, leading to frustration	Smooth and secure user experience with intuitive features

Table 2.1 Comparison

Existing System:

- **Limited Supervision:** Current online exams lack real-time supervision, relying on students' integrity without active monitoring.
- **Security Gaps:** Without advanced measures, cheating is possible, as proctors have limited means to detect or prevent it during exams.

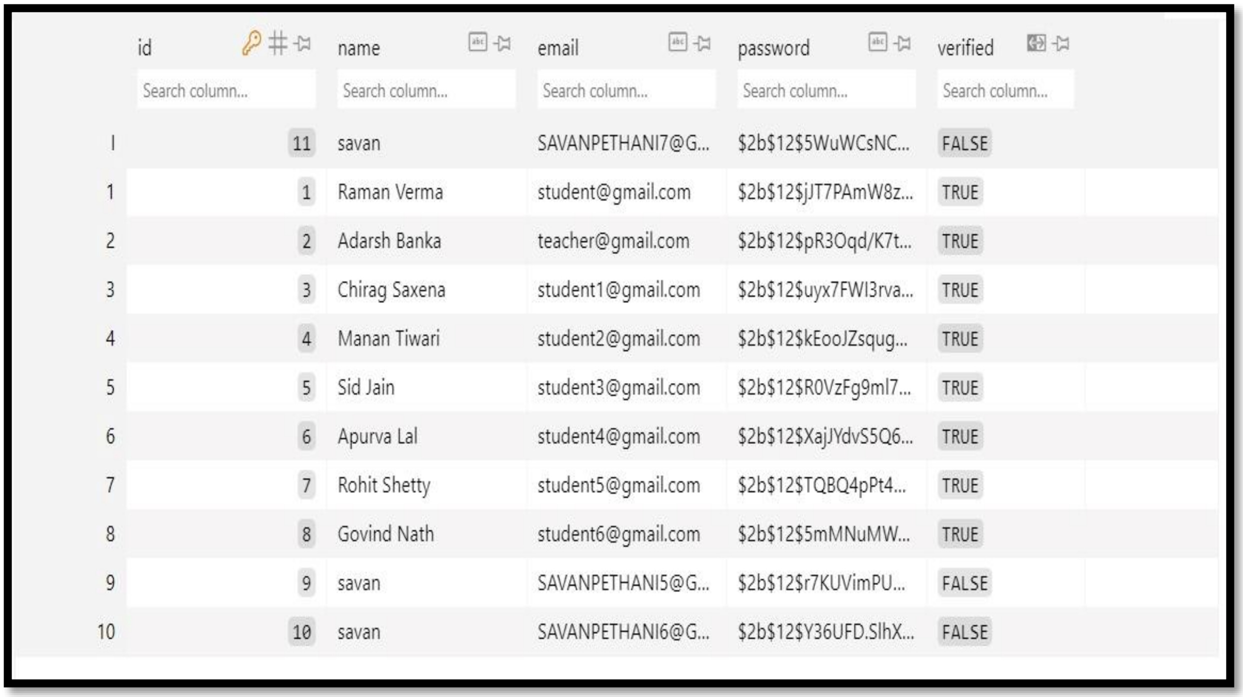
- **Communication Challenges:** Communication between proctors and students is slow and asynchronous, often through emails or messaging platforms, leading to delays in issue resolution.

Proposed System:

- **Active Monitoring:** The proposed system incorporates AI-driven technology to actively monitor exams, detecting suspicious behaviors in real-time.
- **Enhanced Security:** With live monitoring, proctors can swiftly intervene if any anomalies are detected, ensuring the integrity of the examination process.
- **Real-Time Communication:** Instant chat functionality enables immediate communication between proctors and students, facilitating quick assistance and query resolution during exams.

2.2. Database Design

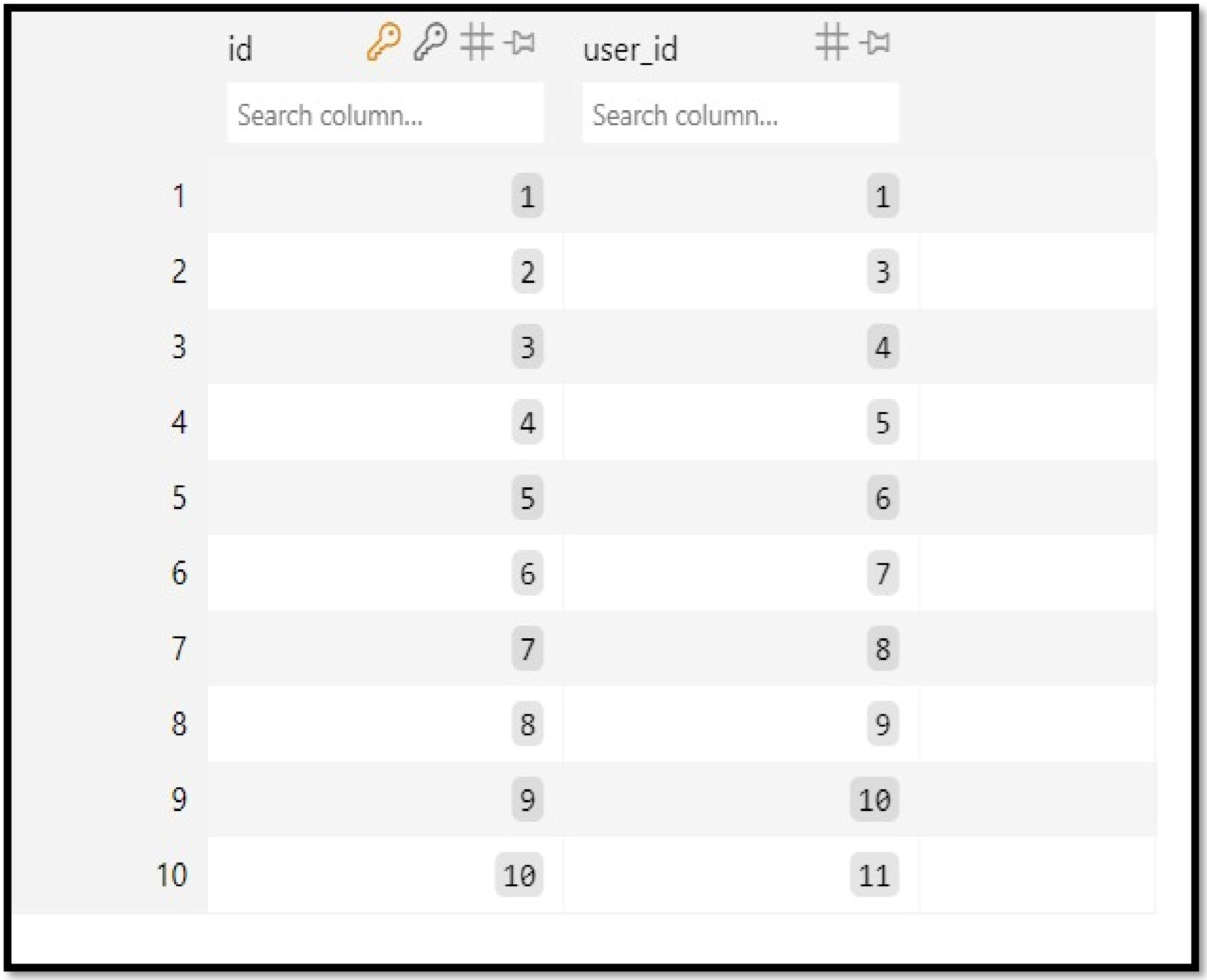
2.2.1. Users



	id	name	email	password	verified
	Search column...	Search column...	Search column...	Search column...	Search column...
I	11	savan	SAVANPETHANI7@G...	$2b$12$5WuWCsNC...	FALSE
1	1	Raman Verma	student@gmail.com	$2b$12$jJT7PAmW8z...	TRUE
2	2	Adarsh Banka	teacher@gmail.com	$2b$12$pR3Oqd/K7t...	TRUE
3	3	Chirag Saxena	student1@gmail.com	$2b$12$uyx7FWI3rva...	TRUE
4	4	Manan Tiwari	student2@gmail.com	$2b$12$kEooJZsqug...	TRUE
5	5	Sid Jain	student3@gmail.com	$2b$12$R0VzFg9ml7...	TRUE
6	6	Apurva Lal	student4@gmail.com	$2b$12$XajJYdvS5Q6...	TRUE
7	7	Rohit Shetty	student5@gmail.com	$2b$12$TQBQ4pPt4...	TRUE
8	8	Govind Nath	student6@gmail.com	$2b$12$5mMNuMW...	TRUE
9	9	savan	SAVANPETHANI5@G...	$2b$12$r7KUVimPU...	FALSE
10	10	savan	SAVANPETHANI6@G...	$2b$12$Y36UFD.SlhX...	FALSE

Fig. 2.2. 1 Users Database

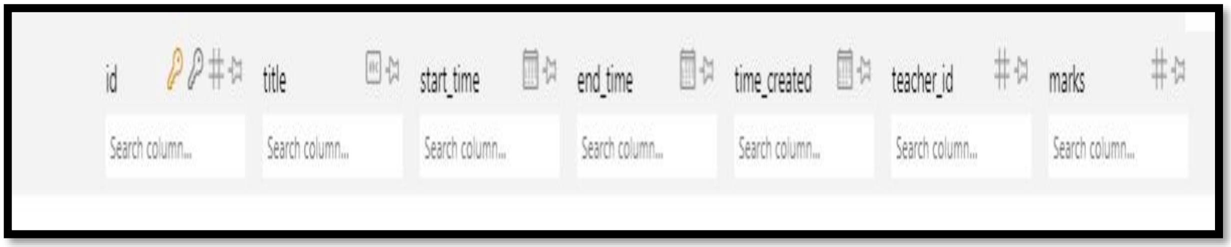
2.2.2. Student



	id	user_id
1	1	1
2	2	3
3	3	4
4	4	5
5	5	6
6	6	7
7	7	8
8	8	9
9	9	10
10	10	11

Fig. 2. 2.2 Student Database

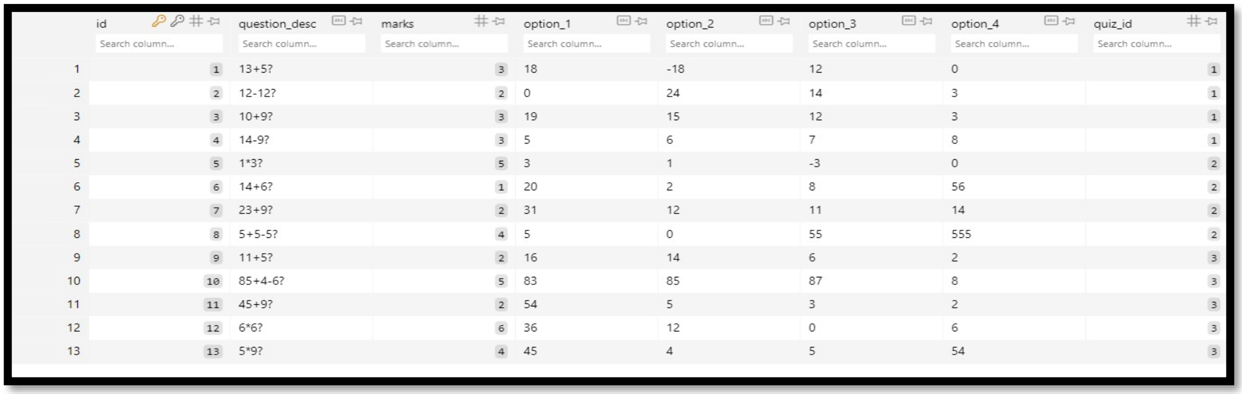
2.2.3. Assignment



id	title	start_time	end_time	time_created	teacher_id	marks

Fig. 2.2. 3 Assignment Database

2.2.4. Quiz



	id	question_desc	marks	option_1	option_2	option_3	option_4	quiz_id
	Search column...	Search column...	Search column...	Search column...	Search column...	Search column...	Search column...	Search column...
1	1	13+5?	3	18	-18	12	0	1
2	2	12-12?	2	0	24	14	3	1
3	3	10+9?	3	19	15	12	3	1
4	4	14-9?	3	5	6	7	8	1
5	5	1*3?	5	3	1	-3	0	2
6	6	14+6?	1	20	2	8	56	2
7	7	23+9?	2	31	12	11	14	2
8	8	5+5-5?	4	5	0	55	555	2
9	9	11+5?	2	16	14	6	2	3
10	10	85+4-6?	5	83	85	87	8	3
11	11	45+9?	2	54	5	3	2	3
12	12	6*6?	6	36	12	0	6	3
13	13	5*9?	4	45	4	5	54	3

Fig. 2.2. 4 Quiz Database

2.3.System Design

- System Architecture:

  The application follows a client-server architecture, where the client-side is responsible for presenting the user interface and handling user interactions, while the server-side manages data processing, business logic, and communication with external services. The architecture consists of the following components:

  - Client-Side Components: HTML, CSS, and JavaScript are used for building the frontend user interface, which interacts with the server through HTTP requests. Client-side components include UI elements for exam navigation, chat functionality, and video streaming.
  - Server-Side Components: The backend of the application is developed using Flask framework in Python. It handles incoming requests from clients, processes data, communicates with the database, and integrates AI models for suspicious activity detection. The server also facilitates real-time communication between proctors and students via WebSockets.
  - Database: The system utilizes a relational database management system (RDBMS) such as SQLite or PostgreSQL for storing user data, exam details, chat transcripts, and other relevant information. The database schema is designed to ensure data integrity, consistency, and efficient querying.

- Database Schema:

  The database schema comprises multiple tables to store different types of data related to users, teachers, quizzes, and quiz questions. Example tables include:

  - Users: Stores information about registered users, including their username, password hash, role (student or teacher), and contact details.
  - Teachers: Contains details about registered teachers, such as their name, email, and unique identification.
  - Quizzes: Stores information about scheduled quizzes, including quiz ID, title, date, time, duration, and associated teacher ID.
  - QuizQuestions: Contains details about questions included in each quiz, such as question ID, content, options, correct answer, and associated quiz ID.

- Integration of AI Models:

  The system integrates AI models for detecting and preventing suspicious activities during quizzes, such as cheating or impersonation. OpenCV is used for real-time analysis of video streams to identify anomalies or prohibited actions, while TensorFlow or PyTorch may be employed for training deep learning models to recognize patterns of suspicious behavior. The AI models are integrated into the backend server and run continuously during quizzes to monitor and alert proctors of any detected irregularities.

- Communication Protocols:

  Real-time communication between teachers and students is facilitated through WebSockets, allowing for instant messaging and issue resolution during quizzes. The server manages WebSocket connections and relays messages between clients, enabling seamless communication without the need for page refreshes.

2.4.Structural System Analysis

2.4.1. Data Flow Diagram

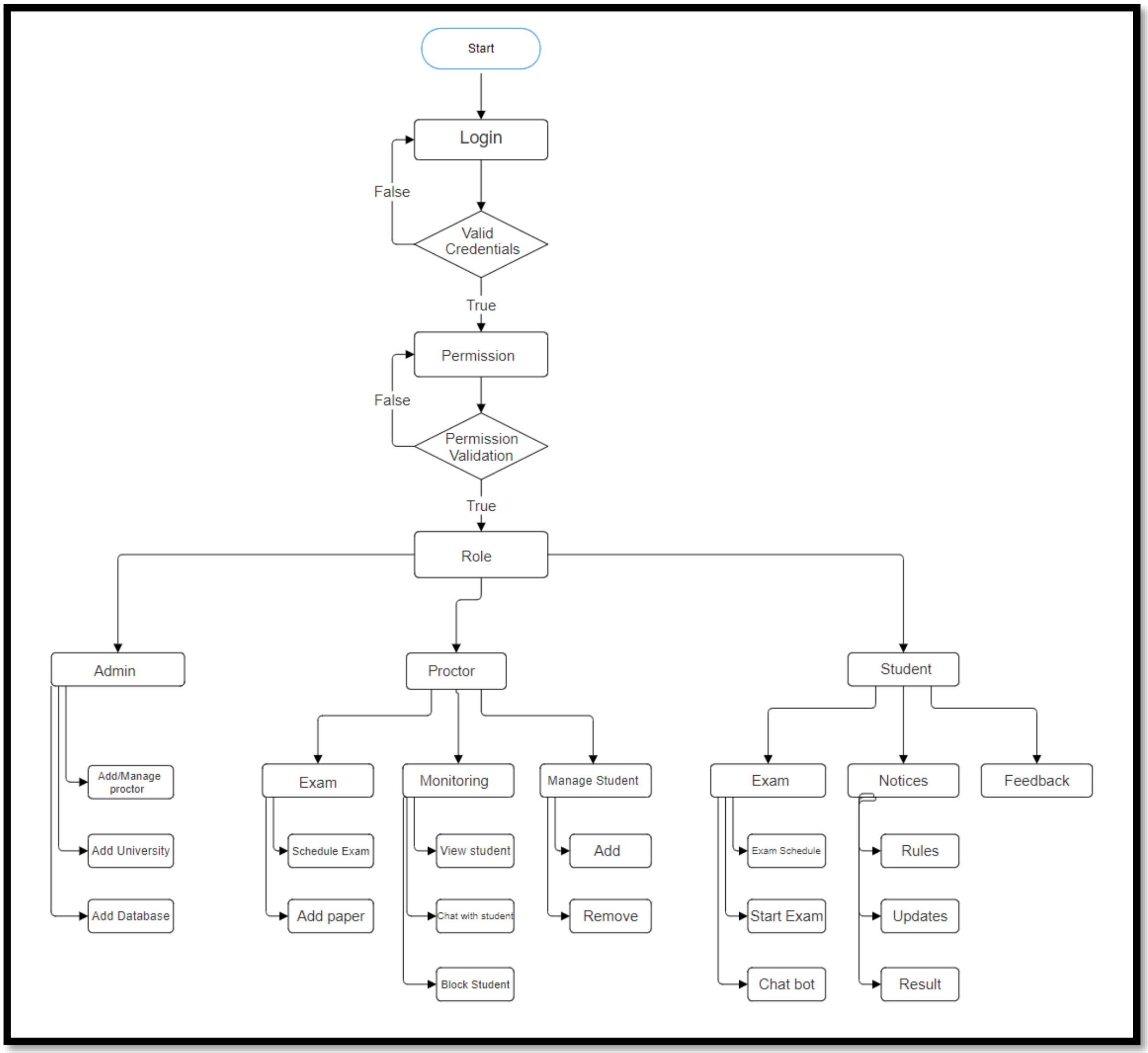


Fig. 2.4. 1 Data Flow Diagram

The data flow diagram for the Project Online Proctor Exam System illustrates the flow of information between different components such as users, exam questions, database, and proctoring tools. It visually represents how data moves from the user input stage through validation, processing, and storage, facilitating a comprehensive understanding of the system's functionality and data interactions.

2.4.2. Entity Relation Diagram

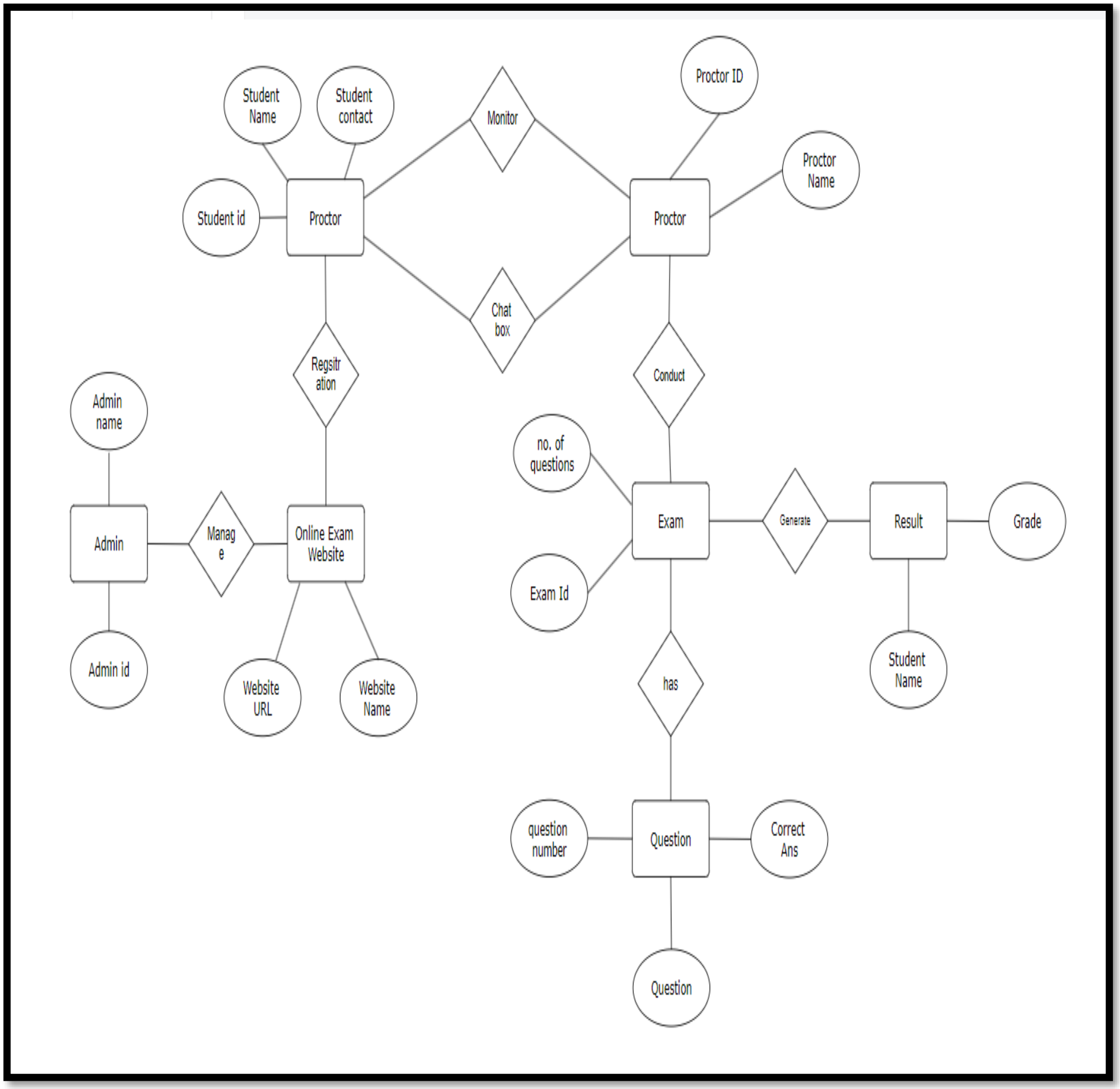


Fig. 2.4. 2 Entity Relation Diagram

The Entity-Relationship Diagram (ERD) for the Project Online Proctor Exam System outlines the various entities involved, such as users, exams, questions, proctors, and the relationships between them. This graphical representation helps in understanding the structure of the database, including entity attributes and their connections, facilitating efficient system design and data

2.4.3. Control Flow Diagram

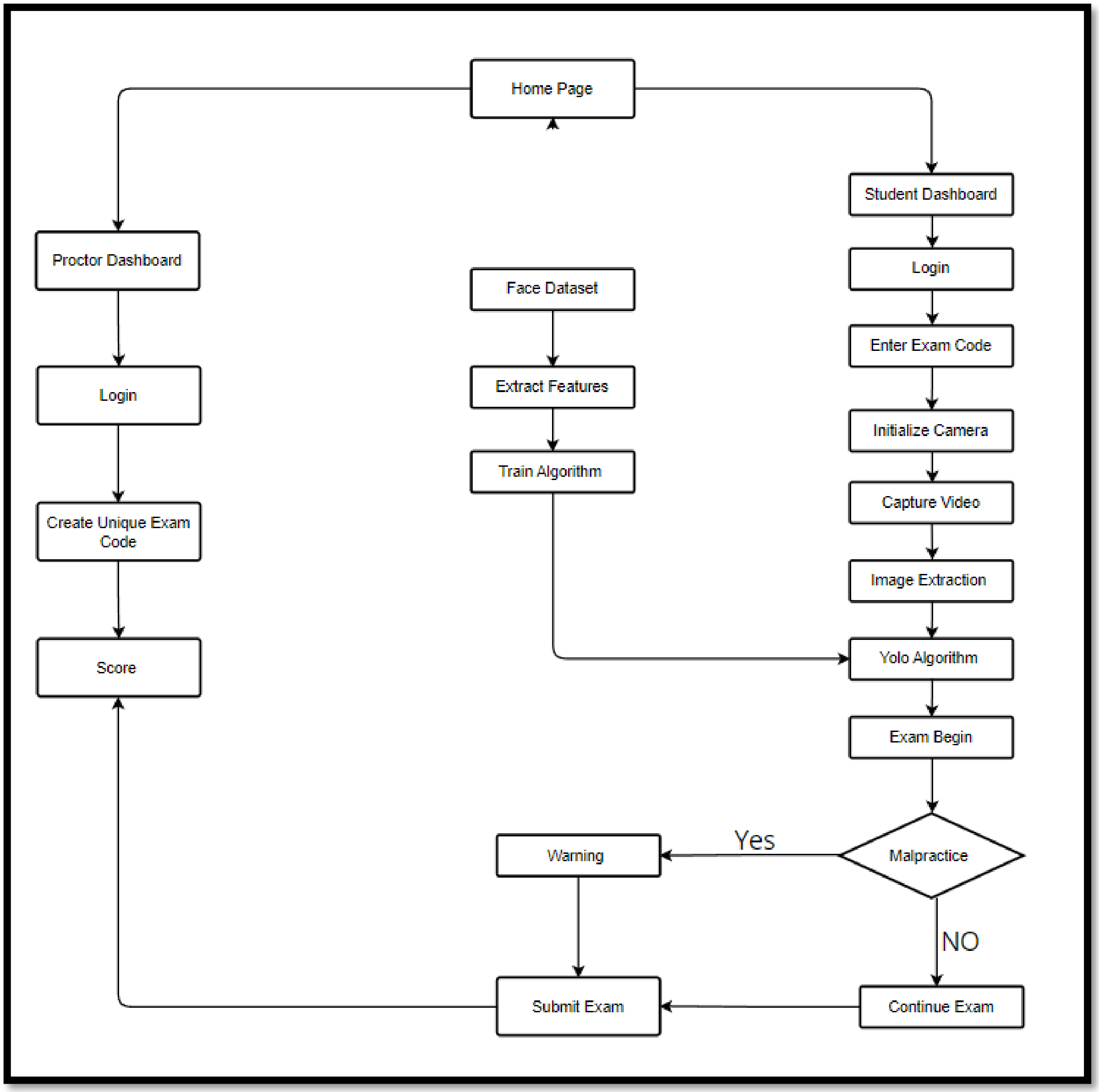


Fig. 2.4. 3 Control Flow Diagram

The Control Flow Diagram for the Project Online Proctor Exam System demonstrates the sequence of activities and decisions involved in the exam process, including user authentication, exam selection, question answering, and proctoring. It visually represents the control flow of the system, highlighting the decision points, loops, and conditional branches, aiding in the understanding and analysis of system behavior and logic flow.

2.5. Object Oriented Analysis and Design

2.5.1. Activity Diagram

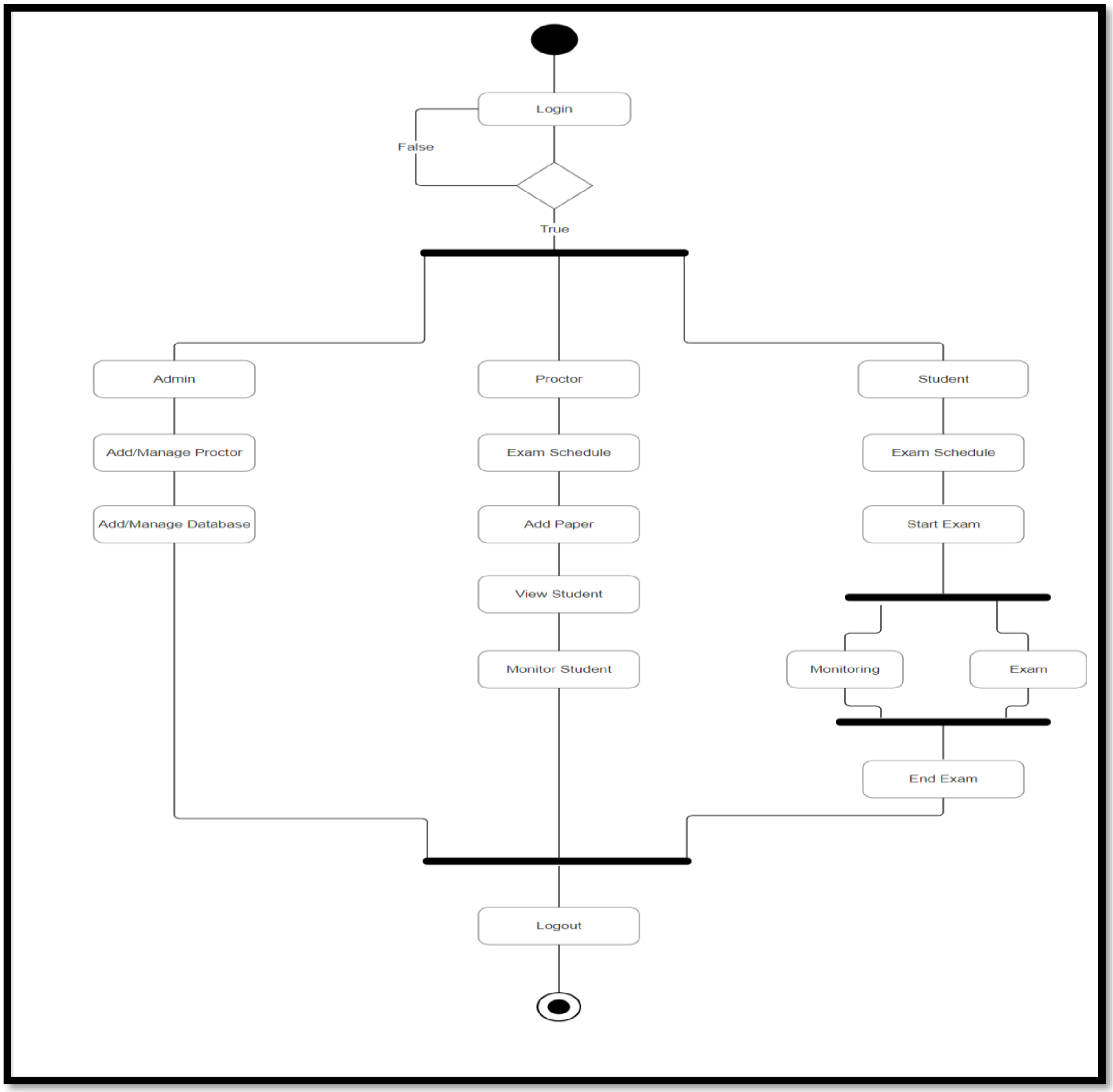


Fig. 2.5. 1 Activity Diagram

The Activity Diagram for the Project Online Proctor Exam System illustrates the sequential flow of activities involved in the exam process, such as user login, exam selection, question answering, and submission. It visually represents the actions, decisions, and transitions between different states, providing a comprehensive overview of the system's functionality and user interactions.

2.5.2. Sequence Diagram

Sequence Diagram Login :

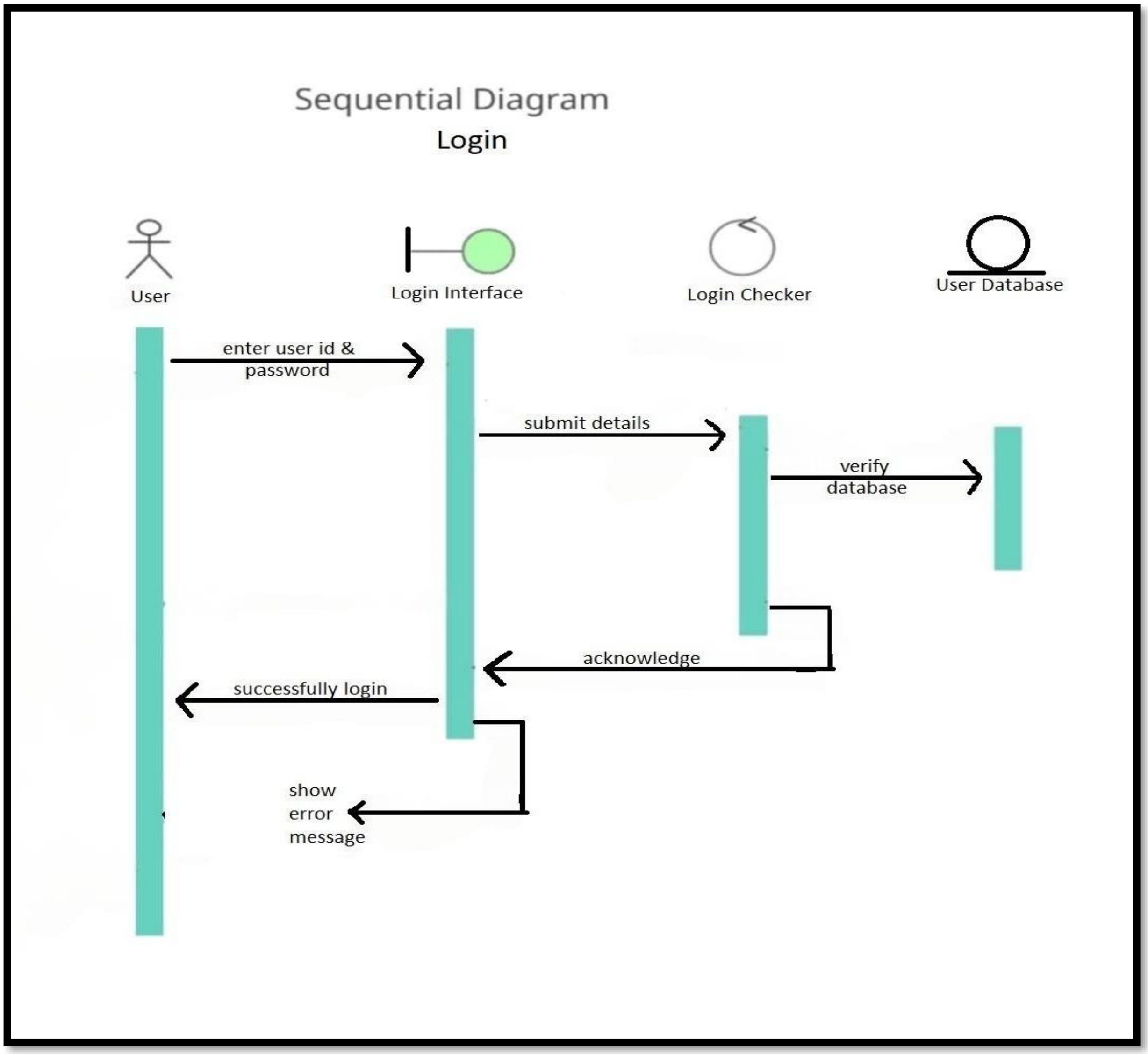


Fig. 2.5. 2 Sequence Diagram Login

In the Sequence Diagram for the login process of the Project Online Proctor Exam System, it begins with the user initiating the login request. The system then verifies the user's credentials against the database. If the credentials are valid, the system grants access and sends a success response, allowing the user to proceed with the exam selection. If the credentials are invalid, an error message is sent back to the user indicating the failure of the login attempt.

Sequence Diagram Permission:

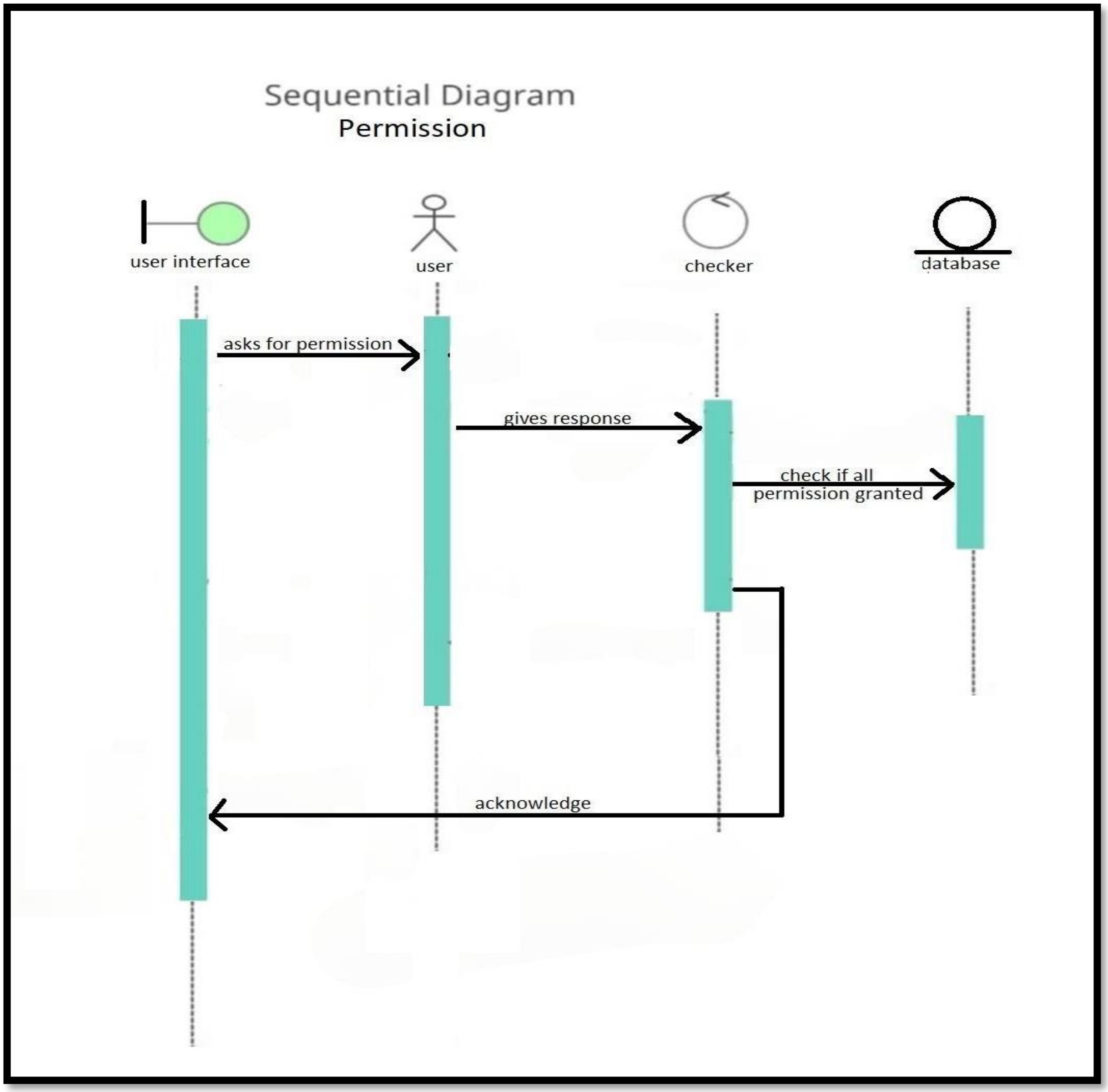


Fig. 2.5. 3 Sequence Diagram Permission

In the Sequence Diagram for permission management in the Project Online Proctor Exam System, it starts with the administrator initiating a request to modify user permissions. The system then verifies the administrator's credentials and authorization level. If the administrator has sufficient privileges, the system proceeds to update the permissions accordingly and sends a success response. If the administrator lacks the necessary privileges or credentials are incorrect, an error message is returned, indicating the failure to modify permissions.

Sequence Diagram Exam:

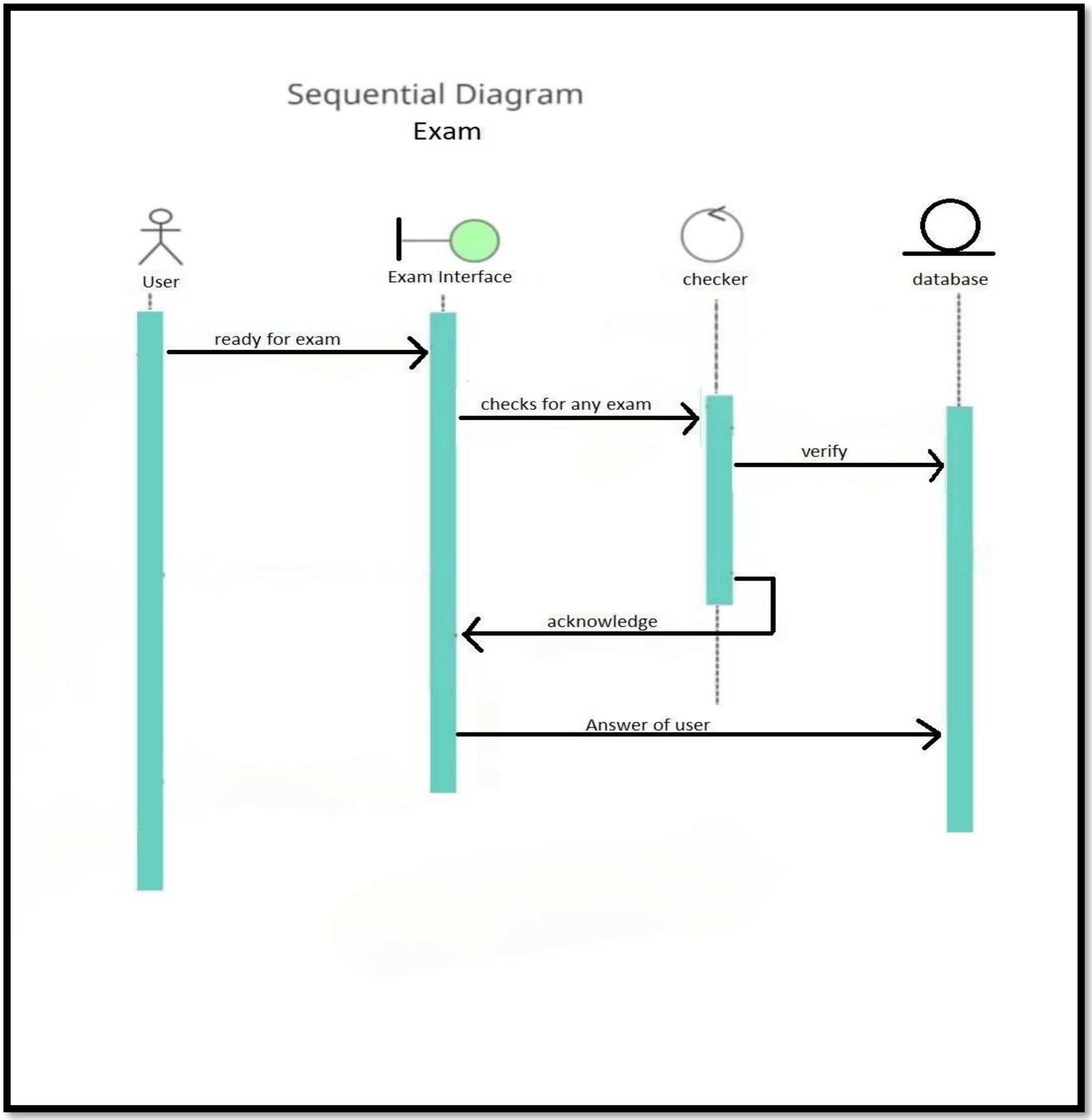


Fig. 2.5. 4 Sequence Diagram Exam

In the Sequence Diagram for the exam process in the Project Online Proctor Exam System, it commences with the user selecting an exam to attempt. The system validates the user's eligibility and exam availability. If the user meets the criteria, the system presents the exam interface. The user then progresses through the exam, answering questions sequentially. As the user completes each question, the system verifies the answer and progresses to the next question until the exam is finished. Upon completion, the system calculates the score and presents the result to the user.

Sequence Diagram Database:

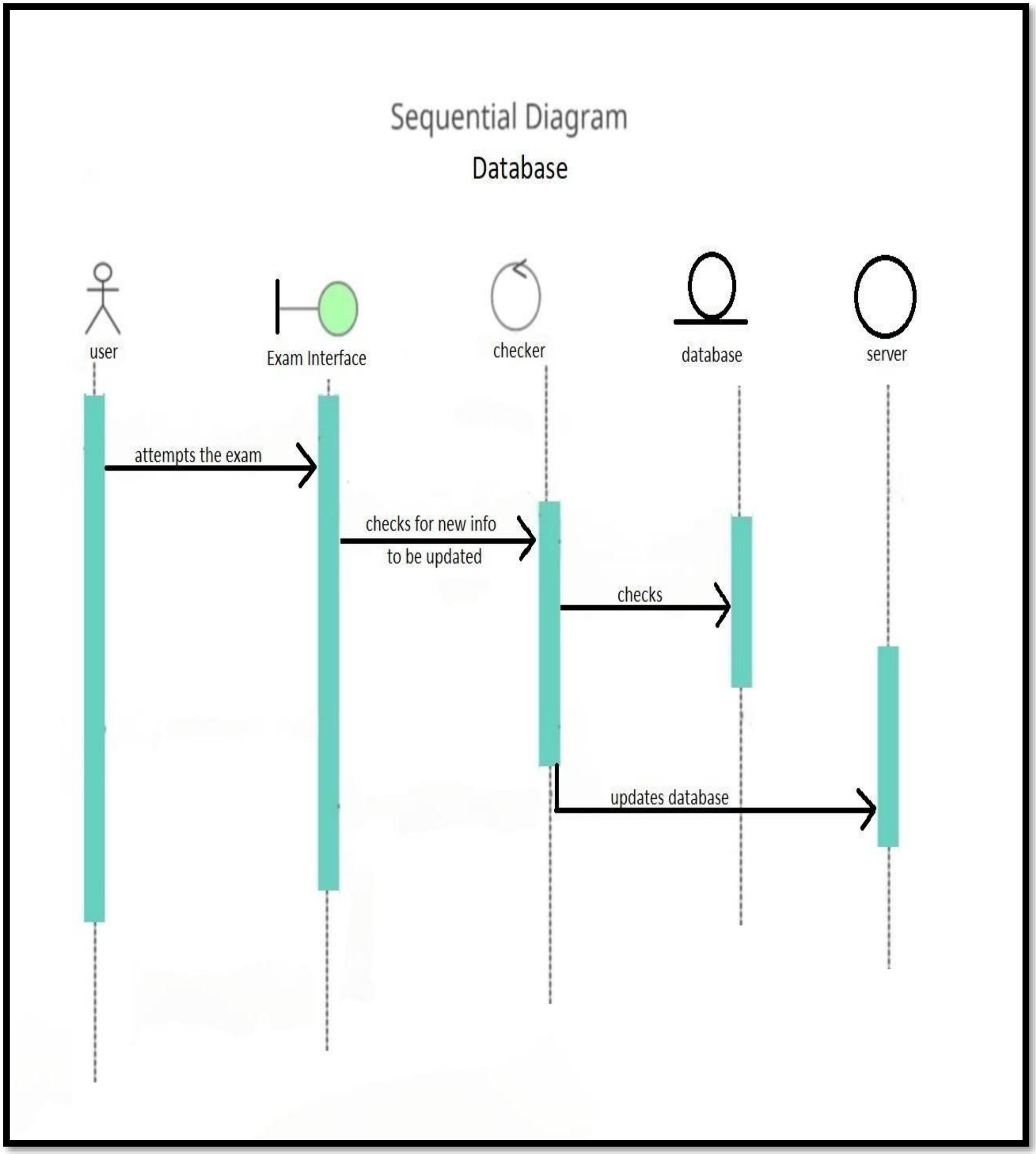


Fig. 2.5. 5 Sequence Diagram Database

The user initiates exam registration, prompting the system to validate information, query the database for available exams, and upon selection, save registration details in the database, confirming the process back to the user.

Sequence Diagram Chatbox:

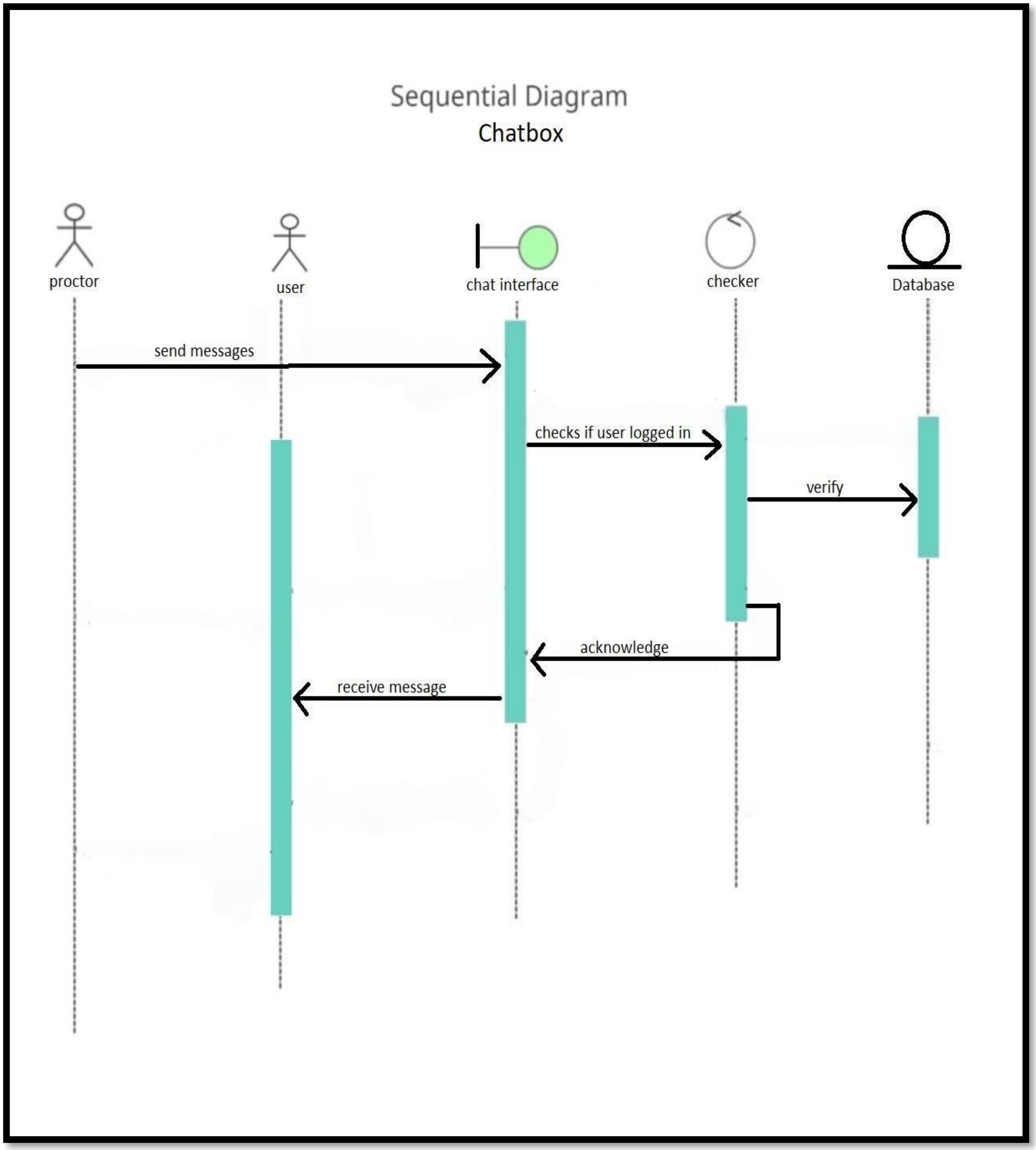


Fig. 2.5. 6 Sequence Diagram Chatbox

The user inputs a message in the chatbox, which the system processes and forwards to the recipient(s); upon receipt, recipients may respond, triggering further updates to the chatbox interface.

2.5.3. Class Diagram

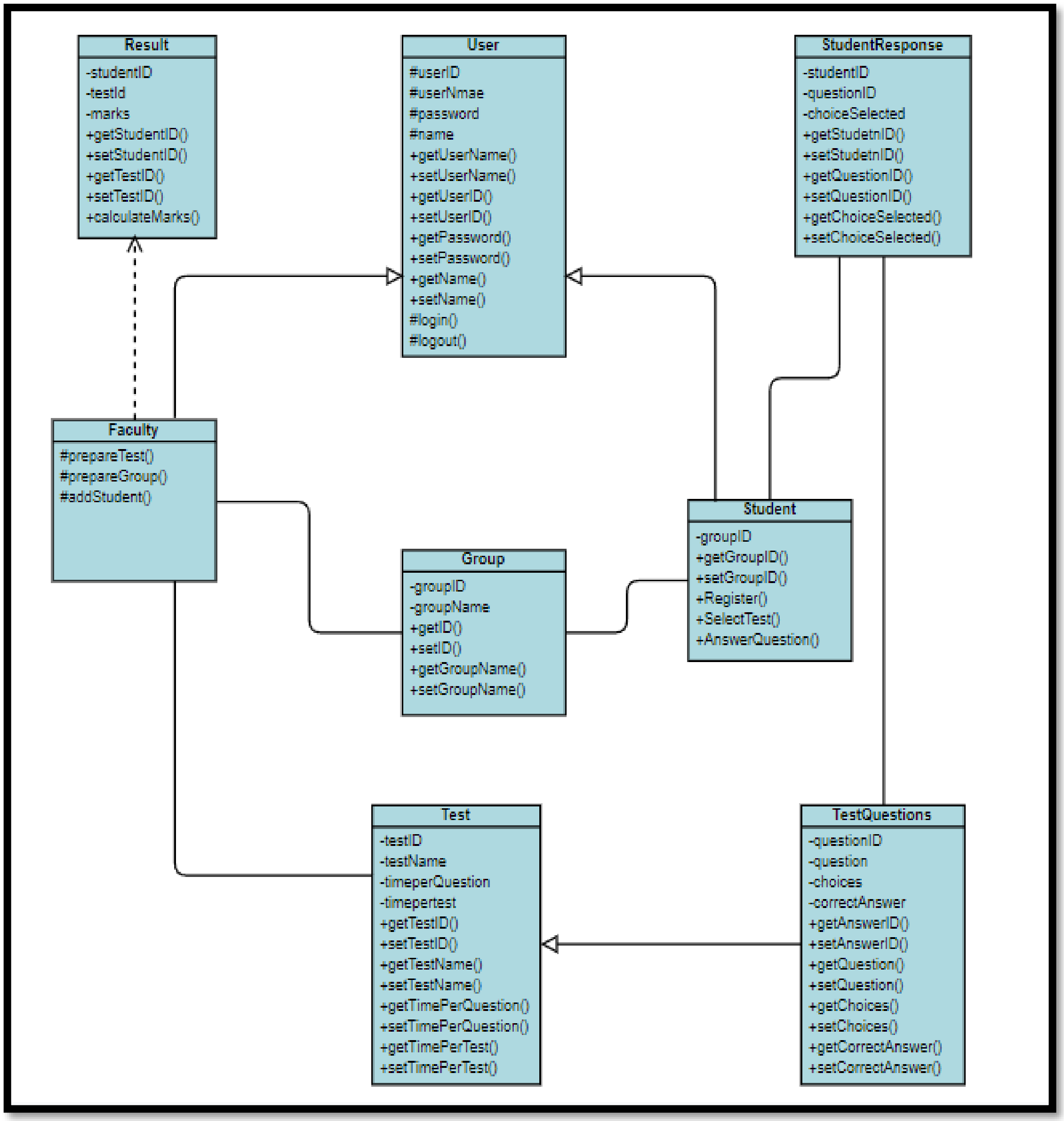


Fig. 2.5. 7 Class Diagram

The Class Diagram for the Project Online Proctor Exam System outlines the various classes such as User, Exam, Question, Proctor, and their relationships, encapsulating attributes and methods within each class to represent the system's structure and functionality. It provides a visual representation of the system's object-oriented design, illustrating how different classes interact and collaborate to facilitate exam management and proctoring processes.

2.5.4. State Diagram

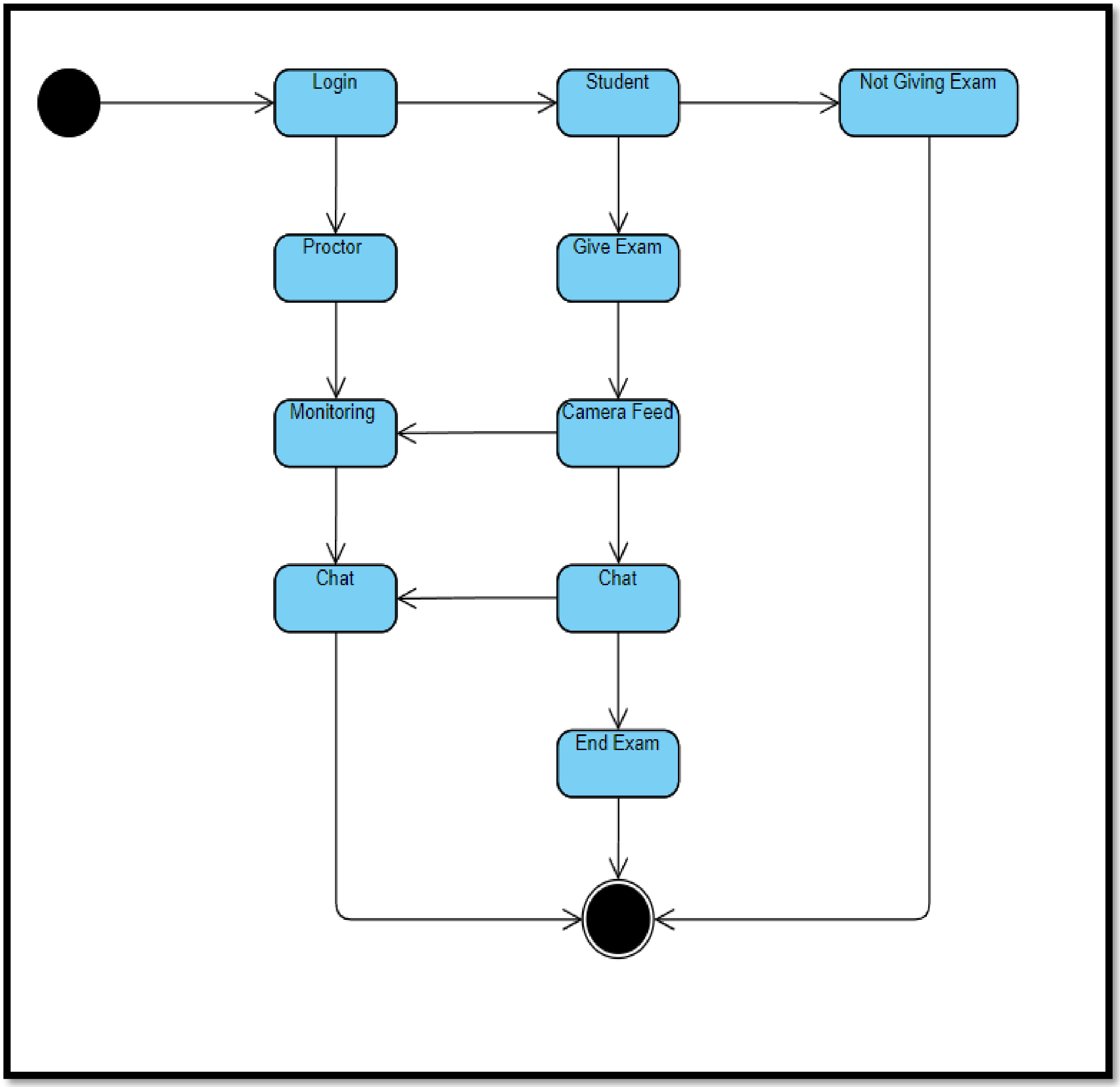


Fig. 2.5. 8 State Diagram

The State Diagram for the Project Online Proctor Exam System illustrates the various states that entities such as users, exams, and questions can undergo, including initial, active, and completed states, along with the transitions between them based on user interactions and system events. It provides a visual representation of the dynamic behavior of the system, helping to understand how entities evolve and progress throughout the exam process.

2.5.5. Use Case Diagram

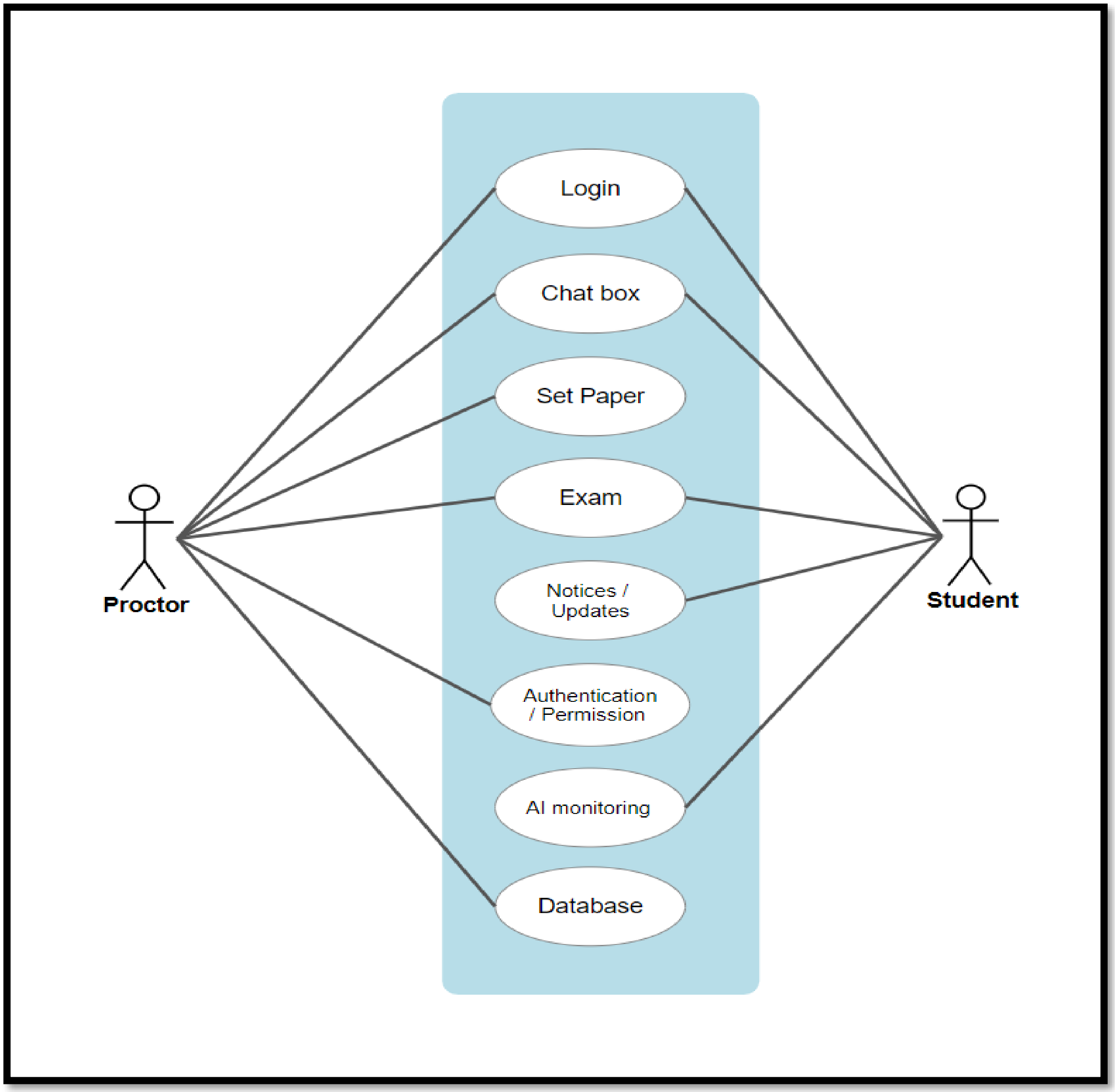


Fig. 2.5. 9 Use Case Diagram

The Use Case Diagram for the Project Online Proctor Exam System depicts the various actors such as students, proctors, and administrators, along with the actions they can perform, such as taking exams, monitoring exams, and managing system settings. It provides a visual overview of the system's functionality from the perspective of different users, aiding in understanding the interactions between users and the system.