SRS Document (Object Oriented Analysis)

1. Introduction

1.1. Purpose

With the current pandemic situation, all educational institutions are converting their examination pattern to be online tests. So it is necessary to manage students online to make the remote exams fair.

This software aims to build a cheating detection system to improve the standards of online exams, ease the load on the teachers, and ensure exams' integrity by confirming the student's identity and monitoring him/her through a webcam.

This document is intended for both the stakeholders and the developers of the system. It describes the system's scope, functional and non-functional requirements for the software, design constraints, and system interfaces.

1.2. Scope

The software (Cheating Detection System) enables students to write exams remotely while maintaining exams' integrity. The software will recognize the student attempting the exam and continuously monitor him/her till the end of the exam and log suspicious activities into a folder. The activities include constantly analyzing the video for any mobile phone, another person, eye movements, etc. This will reduce the human resources required to manage students during exams and improve the proctoring mechanism beyond human-level accuracy.

This SRS describes only the required functionalities of CDS, not the functionality of external systems like data storage, version control systems, or any other frameworks.

1.3. Definitions, Acronyms and Abbreviations

CDS	Cheating detection system
DB	A database where all the data is stored
User	Teacher and student who accesses the system
Teacher	A type of user who can create exams.
Student	A kind of user who attempts exams.

1.4. References

3.1.1 7 things you should know about remote proctoring
3.2.1 Remote proctoring
3.3.1 Understanding Object Detection

1.5. Overview of Document

The next chapter, the Overall Description section, gives an overview of the functionality of the product. It describes the informal requirements and establishes a context for the technical requirements specification in the next chapter.

The third chapter, Specific Requirements, is written primarily for the developers and describes the product's functionality in technical terms.

Both sections of the document describe the same software product in its entirety but are intended for different audiences and use different languages.

2. Overall Description

This section will provide a high-level overview of the entire system. The system will be discussed in its context to demonstrate how it interacts with other systems and to demonstrate its fundamental functions. It will also specify the types of stakeholders who will use the system and the available features. Finally, the system's restrictions and assumptions will be described.

2.1. Product Perspective

CDS is a system that provides functionality described in the 'Product Functions' section. It includes all subsystems needed to fulfill these software requirements. In addition, the CDS has interfaces to external systems like Version Control System. These interfaces shall be implemented according to the available industry standards and be independent of a specific external system.

Any detailed definition of an external system is out of the scope of this document.

There are only two users for the software. Figure 1 represents the overall view of the system.

The CDS system allows the teacher to create an exam by specifying the time duration, the date of the exam, uploading the questions and securing unauthorized access by creating a security code. Likewise, the student can attempt an exam by specifying the security code and validating his/her face. The software continuously checks for potential risk activities, warns the student and stores the data in a log file. After the exam is completed, the teacher can review the log file of each student to analyze his behavior while allocating marks. Likewise, the students can check their results after the teacher uploads them. Since this is a data-centric product, it will need somewhere to store the data. For that, a database will be used. Both the server and the client will communicate with the database, however, in slightly different ways. The AI system will be deployed on the server and the server will have both read and write access to the DB while the client will have only read access to the DB.

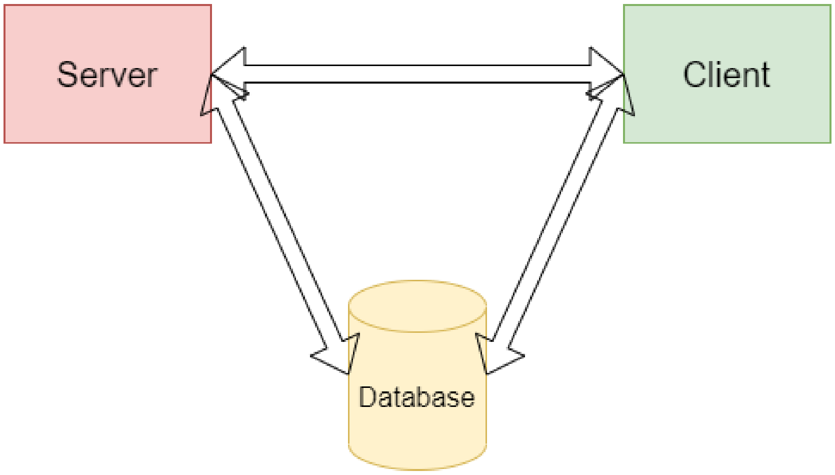


Figure 1: Overall view of the system

2.2. Product Functions

The primary functions of the CDS are:

3.1.1 A user can register or login into the system.
3.2.1 A teacher can create and schedule an exam.
3.3.1 A student can join the exam.
3.4.1 The system validates students based on their facial features.
3.5.1 The system captures suspicious activities and stores them in a log file.

3.6.1 The system automatically corrects the responses with the answers.
3.7.1 The teacher can update the marks by checking the log files.

2.3. User Characteristics

Users of the system should be able to register and login. In addition, the system should validate the users based on their facial features.

The teachers should be able to do the following additional functions:

3.1.1 Create and schedule an exam.
3.2.1 Upload a document of questions.
3.3.1 Set parameters for the exam.
3.4.1 View the log files of different students.
3.5.1 Upload results.

The student should be able to do the following additional functions:

3.1.1 Join an exam.

3.1.2 Write the exam
3.1.3 View the results.

2.4. User Constraints

3.1.1 The camera should be activated only when necessary.
3.2.1 Streaming video on hundreds of devices at a time will increase the load on the database.
3.3.1 AI models are computationally expensive.
3.4.1 The Internet connection is also a constraint for the application. Since the application fetches data from the database over the Internet, it is crucial that there is an Internet connection for the application to function.
3.5.1 The capacity of the database will constrain both the web portal and the mobile application. Since the database is shared between both applications, it may be forced to queue incoming requests and therefore increase the time it takes to fetch data.
3.6.1 HD video capturing requires large bandwidth.

2.5. Assumptions and Dependencies

3.1.1 Users are assumed to have distinguishable facial features to be recognized uniquely by the system.
3.2.1 Users are either employees in the institution or students studying in the institution.
3.3.1 Inform the teacher if a user disconnects in the middle of the exam.
3.4.1 The software is always used in a web browser and not on a mobile device.
3.5.1 The camera is inbuilt into the PC or the laptop.
3.6.1 Users are assumed to have an embedded camera and minimum internet connection to support the system.

2.6. Apportioning Requirements

If the project is delayed, some requirements could be transferred to the next version of the application.

3. Specific Requirements

This section contains all of the functional and quality requirements of the system. In addition, it gives a detailed description of the system and all its features.

3.1. Interface Requirements

This section provides a detailed description of all inputs and outputs from the system. It also describes the hardware, software and communication interfaces and provides basic prototypes of the user interface.

3.1.1 User Interface

If a user is not registered, he/she should register from the register page. If the user is already registered, he/she can login into the system.

Every user should have a profile page where they can edit their e-mail address, profile picture and password.

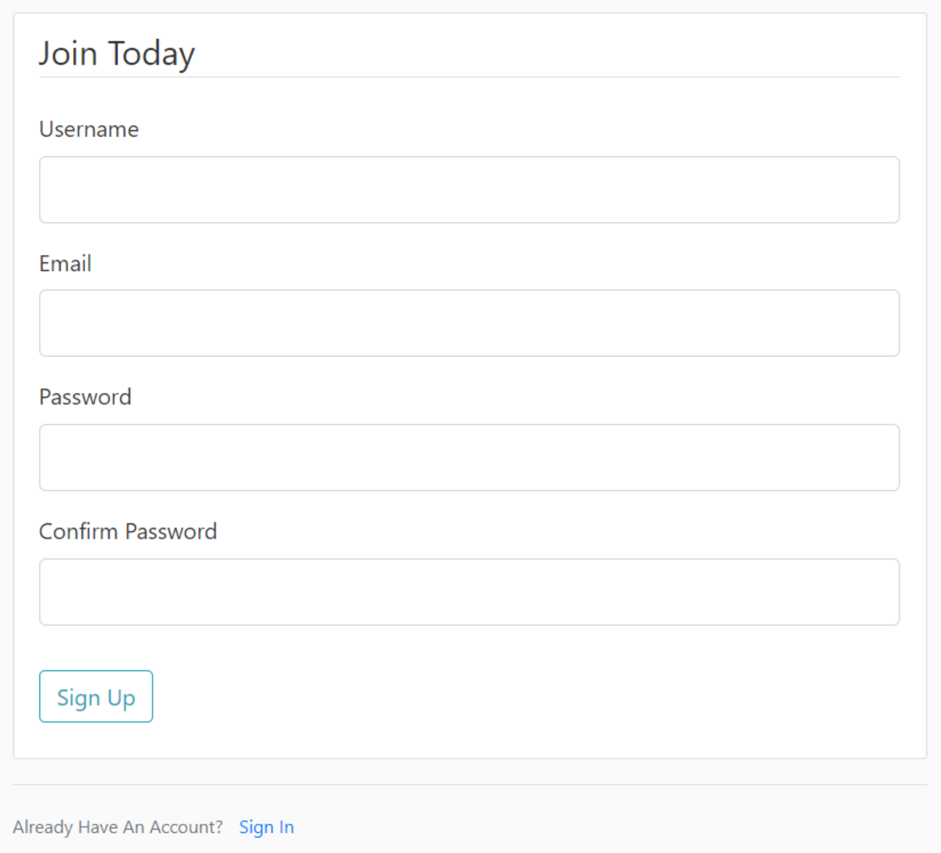


Figure 2: Register page

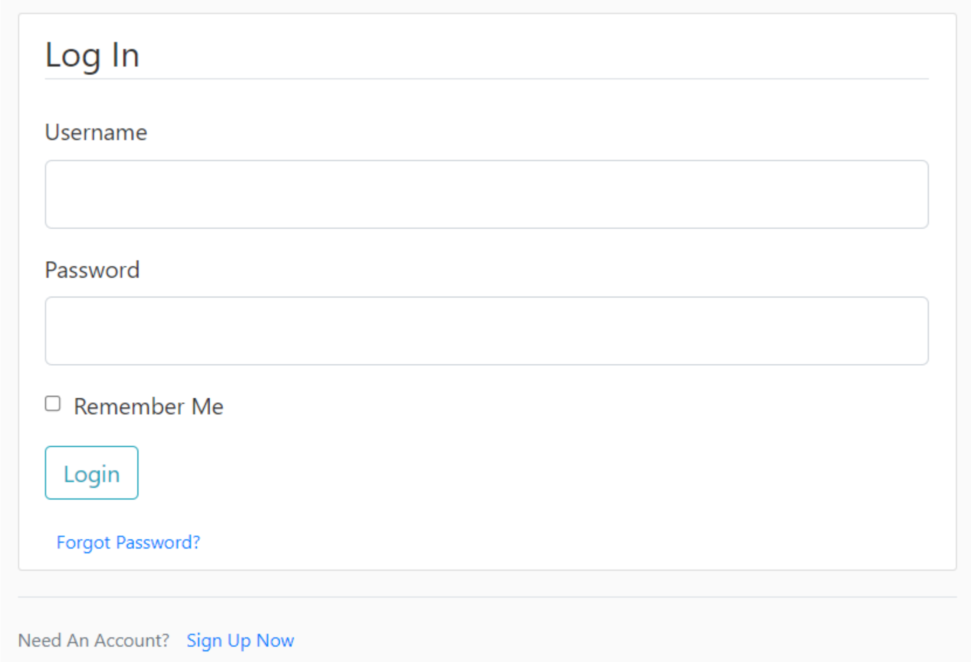
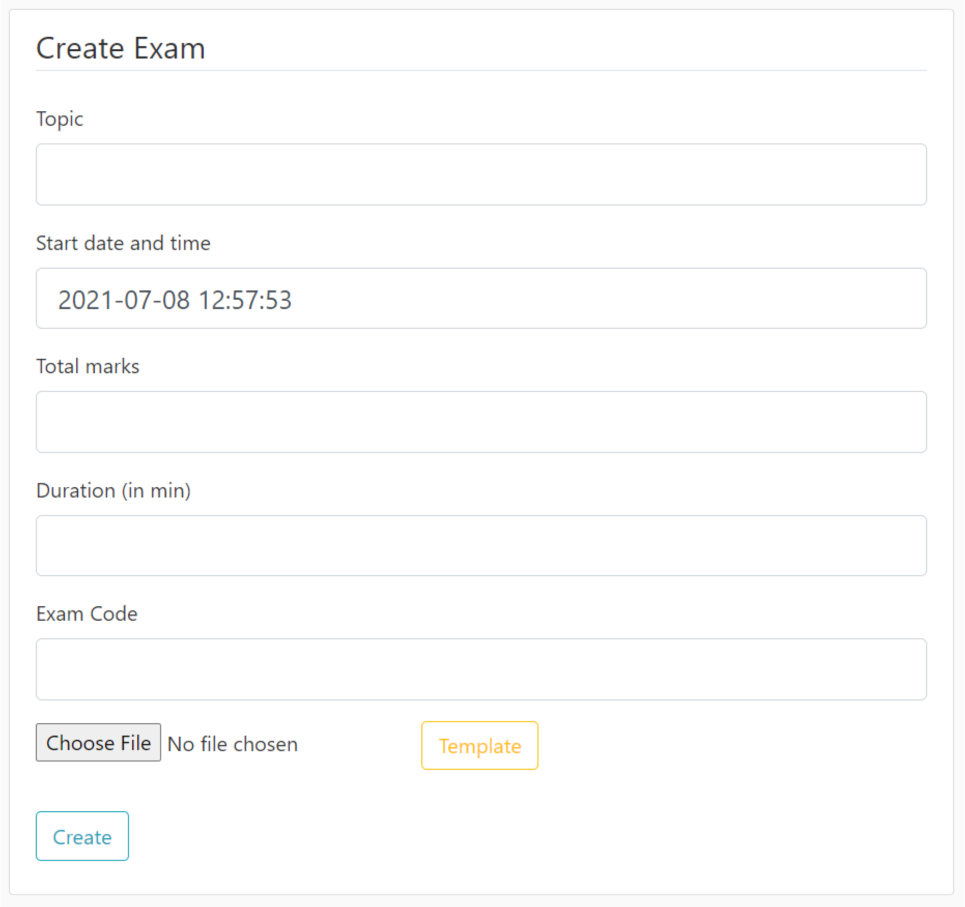


Figure 3: Login page

A user with teacher privileges will be able to create an exam by specifying the topic, the exam duration, the date of examination, a security code to protect the exam from unauthorized users and a file upload option to upload questions. The server will automatically map questions to answers and display the questions to the students. After attempting the exam, it will automatically correct the responses.



Create Exam

Topic

Start date and time

2021-07-08 12:57:53

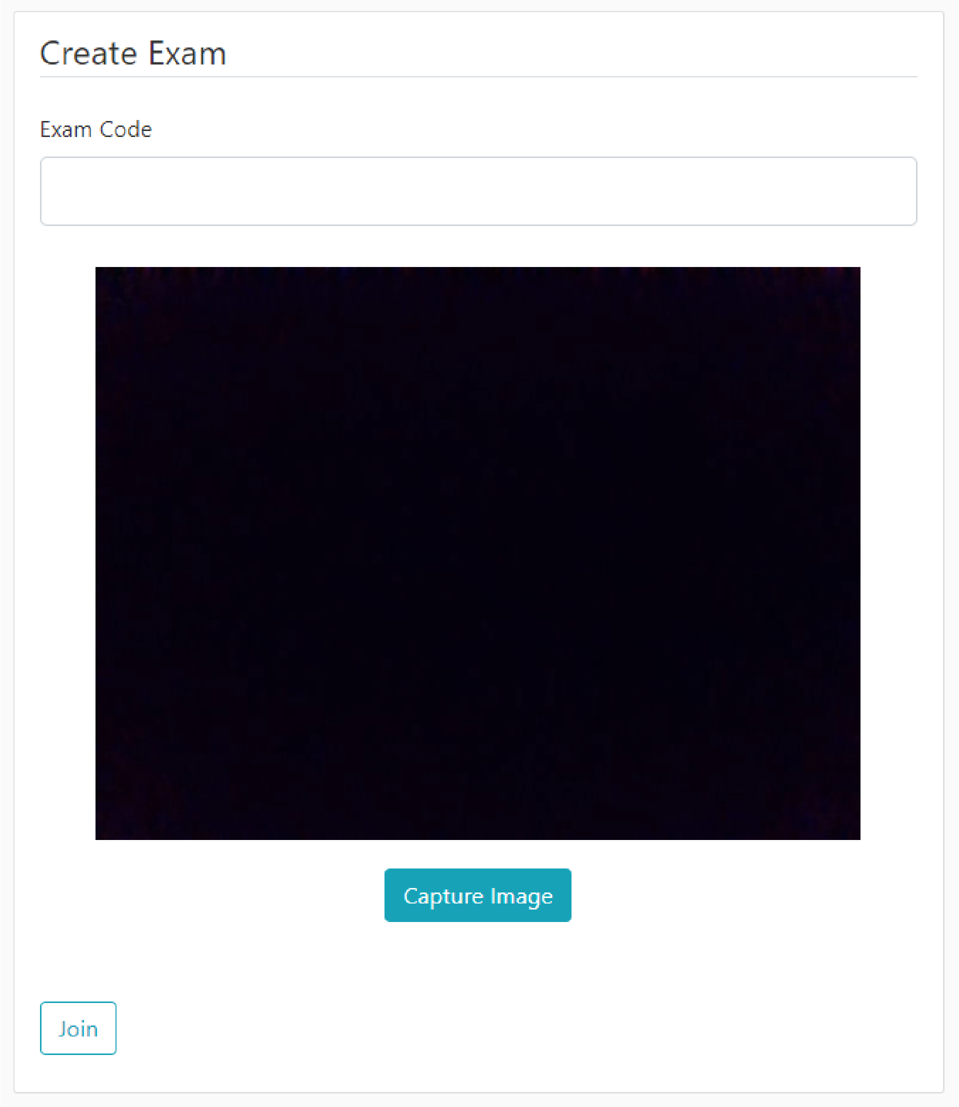
Total marks

Duration (in min)

Exam Code

Figure 4: Exam creation page

A user without teacher access can only attempt exams and view the results. However, they can join exams before the exam starts by providing the security code and successful validation of their faces.



Create Exam

Exam Code

Figure 5: Join exam page

3.1.2 Hardware Interface

Since the client browser does not have any designated hardware, it does not have any direct hardware interfaces. The camera is inbuilt to the laptop or PC.

3.1.3 Software Interface

Both the client and the server communicate with the DB. However, the communication between the DB and the server consists of operations concerning both reading and modifying the data. In contrast, the communication between the database and the client includes only reading operations.

3.1.4 Communication Interface

Communication between the different parts of the system is essential since they depend on each other. However, in what way the communication is achieved is not necessary for the system and is therefore handled by the underlying operating system of the server and the client. The communication makes sure that the system is running smoothly and any abnormalities are informed.

3.2. Functional Requirements

This section includes the requirements that specify all the fundamental actions of the software system.

3.2.1 Use Case Model

Actors	-	student, teacher.
Behavior	-	login, initialize exam, join exam, detect cheating.

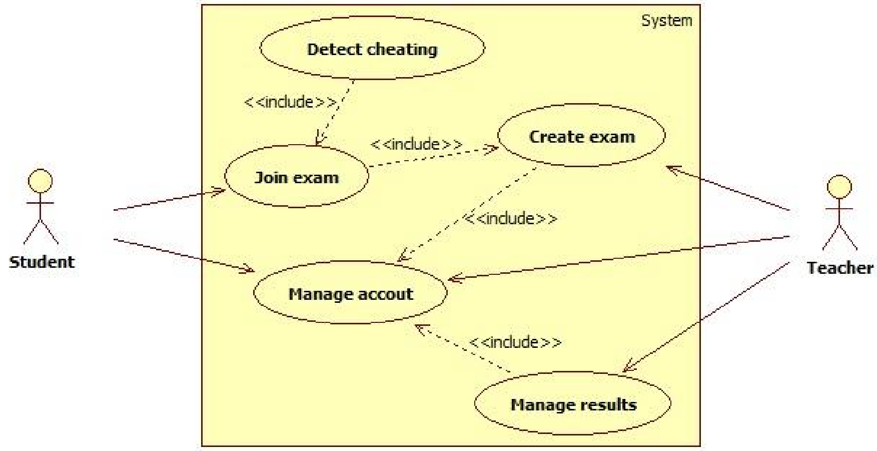


Figure 6: Use case diagram

3.2.2 Use Case Specifications

Refer to Appendix

3.2.3 Activity Diagram

Refer to Appendix

3.2.4 Analysis Classes

Refer to Appendix

Use Case ID	Classes identified (refined after domain analysis)
CD01	Student, Teacher
CD02	Teacher, Exam, Course
CD03	Student, Exam
CD04	Student, Exam
CD05	Teacher, Exam, Result

3.3. Performance Requirements

- The system should be able to take HD frames to detect cheating.
- The face recognition algorithm should be lightweight and the result should be generated within 1 second with at least 90% accuracy.
- The UI should be clean and easy to navigate with minimal design and loaded on the client-side within 1 second.
- The server should be able to serve at least 10 clients in parallel.

3.4. Logical Database Requirements

- The system accepts two types of users - student and teacher. Both types of users have some attributes in common like unique id (integer), name (string), institution (string) and contact number (10-digit integer). Some other attributes differentiate one user from the other. A teacher has the unique privilege to create exams. Every user has to login/register to access the system. So these data are accessed frequently. A user cannot have an empty name, institution, or contact number.
- The exam has some attributes like topic name (string), duration (time), date of examination (date), security code (string) and a data upload option that accepts a document containing questions and answers.
- The system detects cheating and logs the data into a file. The file can be of varying lengths and is stored in the local file system of the server. The file will contain different timestamps of the exam and the predicted activity at that timestamp.

Since a DB is hosted on a dedicated server and the users have restricted access to the DB, the client can only send the data to the server and the server writes the data to the DB. The client can only read from the DB. This ensures that the DB is protected from any unauthorized access.

3.5. Design Constraints

This section includes the design constraints on the software caused by the hardware.

- Server capacity - the server should handle multiple requests at a time and compute complex tasks like detect cheating at once.
- Storage capacity - the database server should be refined periodically as it contains images and data of students, logs which will take up a lot of space.
- Memory management - the memory required to load the system on the client side may be insufficient.
- Internet connectivity - the bandwidth required to run the system may differ based on the activity.
- Hardware availability - to process frames and detect cheating, GPU is needed for parallelization.

3.6. Software System Attributes

3.6.1 Reliability

The system should be able to handle at least 50 students at once and detect cheating in parallel. The system should even scale if two or more exams are being conducted at the same time. The AI should predict with at least 90% accuracy and be trained to predict at 95%+ accuracy.

3.6.2 Availability

The system should backup the data periodically and the writing to DB should be done as transactions (single access at a time) to avoid inconsistency. In case of a network failure (~0.1% of the time), the system should abort all writing access to the DB. The system should be available for at least 99% of the time during testing. The system should be connected to the internet and support video capturing.

3.6.3 Security

User credentials should be encrypted and stored in the database to disallow unauthorized access. A user must register before logging in and login before attempting to write an exam or create an exam. If a student tried to create an exam or pretend to be a teacher, an error message should be displayed. The client should not be allowed to write to DB directly. The data should be transferred using a secured protocol. User’s privacy should be maintained by not storing the video and misusing it. In case of any suspicious activity, a message must be sent to the administrator or a teacher.

3.6.4 Maintainability

The system should easily scale to allow all the students to attempt exams at the same time. The AI models should be trained frequently and the frameworks should be updated to the latest stable versions. Testing should be done on a copy of the DB without disturbing the original DB.

3.6.5 Portability

The system should be portable with any operating system supporting a browser on the client-side. All the elements should be tested on different platforms like Chrome, Firefox, Safari, etc. The server-side should be installed on a specific operating system to manage file accessing, frameworks and dependencies.

4. Supporting Information

4.1. Table of contents

3.6.3. Security

3.6.4. Maintainability

3.6.5. Portability

4. Supporting Information

4.2. Appendix

All the use case diagrams and specifications are given in another document.