A black text with white background

Description automatically generatedA logo with text on it

Description automatically generated

**Examination System project**

**Track: Full stack Development**

**Supervisor: Eng.Rami Nagi**

**By: Eslam Salah**

**Kareem Nasser**

**Omar Dyaa**

**Ahmed Lotfi**

**Aya Mahmoud**

Table of Contents

[Chapter one: Project Scope 3](#_Toc189325821)

[Abstract: 3](#_Toc189325822)

[Problem Definition: 4](#_Toc189325823)

[Deliverables: 5](#_Toc189325824)

[Project constraint: 5](#_Toc189325825)

[Chapter Two: Requirements And Analysis 6](#_Toc189325826)

[1. Project Requirement 6](#_Toc189325827)

[1.1 . Functional Requirements: 6](#_Toc189325828)

[1.2. Functional Requirements: 8](#_Toc189325829)

[1.3. Hardware Requirements: 8](#_Toc189325830)

[1.4. Software Requirements: 8](#_Toc189325831)

[3. Project Diagrams 9](#_Toc189325832)

[3.1. ERD 9](#_Toc189325833)

[3.2. Mapping 10](#_Toc189325834)

[Chapter Three: implementation: 12](#_Toc189325835)

# Tables Of Figures:

[Figure 1: ERD Screenshot 9](#_Toc189325704)

[Pdf 2 ERD File Link 10](#_Toc189325705)

[Figure 3: Mapping diagram. 11](#_Toc189325706)

[Pdf 4: Mapping diagram file link. 11](#_Toc189325707)

[Figure 5;Exam Generation SP part1 13](#_Toc189325708)

[Figure 6:Exam Generation SP part 2 14](#_Toc189325709)

[Figure 7:Student Model Answers SP 15](#_Toc189325710)

[Figure 8: Exam Correction SP Part1 15](#_Toc189325711)

[Figure 9:Exam correction SP part2 16](#_Toc189325712)

[Figure 10:1st Report SP 17](#_Toc189325713)

[Figure 11:2nd Report SP 17](#_Toc189325714)

[Figure 12:3rd Report SP 18](#_Toc189325715)

[Figure 13:4rt Report SP 19](#_Toc189325716)

[Figure 14:5th Report SP 19](#_Toc189325717)

[Figure 15:6th Report SP 20](#_Toc189325718)

[Figure 16: Exam Question Report dashboard 21](#_Toc189325719)

[Figure 17: Student by department and student grades Report dashboard 21](#_Toc189325720)

[Figure 18Courses Topics Report dashboard 22](#_Toc189325721)

# Chapter one: Project Scope

## Abstract:

The Examination System is a comprehensive application developed to manage and automate various aspects of the examination process at an ITI (Industrial Training Institute). This system is built using Microsoft SQL Server as the database platform and aims to streamline the process of conducting exams, recording results, and managing student data efficiently and organized.

The system is designed to store essential information such as student details, course data, exam schedules, and results. It provides functionalities to manage student registrations, track attendance, generate exam schedules, and record marks for each subject. Administrators can access various reports, while faculty members can update exam results and view student performance.

The SQL Server database is critical in maintaining data integrity, supporting complex queries for report generation, and ensuring secure storage of sensitive information. The system is optimized for scalability, allowing it to handle increasing data volume and users as needed.

## Problem Definition:

The goal is to develop a comprehensive Examination System that is efficient, effective, flexible, and secure, ensuring seamless operation at all times. Currently, the process of managing examinations, recording results, and generating reports is often carried out manually or through outdated, inefficient systems. This results in several challenges, including data entry errors, delayed report generation, poor management of student records, and difficulties in tracking and analyzing student performance over time. Additionally, these manual processes are time-consuming and prone to human error, which ultimately affects the accuracy and reliability of exam-related data.

In educational institutions such as ITI, the lack of an integrated, automated system hinders the ability to efficiently handle the growing volume of student data, exam schedules, and results. This leads to bottlenecks and inefficiencies, particularly when trying to generate real-time reports or ensure data security.

To address these issues, we aim to design and implement an automated Examination System that not only streamlines the entire examination workflow—from student registration to exam scheduling and result processing—but also securely manages all related data. The system will provide real-time access to critical information, improve accuracy and speed in report generation, and ensure secure storage of sensitive data, while also being scalable to accommodate the future growth of the institution.

## Deliverables:

1. Entity Relational Diagram (ERD).
2. Mapping Diagram.
3. Examination system Database.
4. Stored Procedures
5. SQL Queries.
6. Triggers
7. Exam Generation
8. Exam Corrections
9. Model Answer of Student Iin the Exam.
10. Reports by SSRS and SSDT.
11. Power bi Dashboard of Reports.

## Project constraint:

1. **Time**: The project must be completed within a specified time frame. This timeline includes all phases of development, from requirements gathering and system design to implementation and testing. Meeting deadlines is critical to ensure that the system is deployed on schedule and can be used in the intended academic cycle.
2. Quality: The Examination System must meet predefined quality standards, ensuring that it is reliable, efficient, and user-friendly. The system should be free from major bugs or errors, have minimal downtime, and perform smoothly under normal usage conditions. Delivering the system with the required quality within the specified time is essential for user satisfaction and operational success.

# Chapter Two: Requirements And Analysis

## 1. Project Requirement

### 1.1 . Functional Requirements:

* **Student Registration and Management:**
  + The system should allow students to register and maintain their personal and academic details, including name, ID, course details, and contact information.
  + The system should allow students to update their information when necessary.
  + The system should store and manage records of all registered students.
* **Course and Subject Management**:
  + Training Managers and Admins should be able to create, update, and delete courses and subjects through Full permissions.
  + Each subject should have a unique ID and be associated with a specific course and instructor.
  + The system should allow for associating exams with respective Courses.
* **Exam Scheduling:**
  + The instructor should be able to schedule exams by specifying the exam date, time, and duration.
  + The system should automatically generate an exam schedule for each student based on their enrolled Courses according to every intake track.
* **Exam Registration and Enrollment:**
  + Students should be able to register for exams for the Course they are enrolled in.
  + The system should maintain records of student exam registrations.
* **Report Generation:**
  + The system should allow administrators and organization members to generate various reports, including student exam results, and model answers of every student in the exam.
  + The system should provide summary reports, such as students in every department, Topics of courses, Questions of the exam, and total grades of students in his/ her enrolled courses.
* **Security and Access Control:**
  + The system should provide role-based access, ensuring that only authorized users (e.g., administrators, faculty, students) can access specific functionalities.
  + Students should only have access to their information, while administrators and faculty can view data across the system.
  + The system should implement secure authentication and data encryption to protect sensitive information.
* **Notifications and Alerts:**
  + The system should send notifications and alerts to students about important dates, such as upcoming exams, deadlines for registration, or result announcements.
  + Faculty and administrators should receive alerts regarding exam schedule conflicts or missing result entries.
* **Backup and Data Recovery:**
  + The system should support regular backups of the database to restore data in case of downtime or malfunction that can be caused because of changed circumstances.
* **Audit and Logs:**
  + The system should maintain logs of all actions taken within the system, such as data changes, user logins, and exam result entries (Auditing).
  + Administrators should be able to view these logs for security and auditing purposes.

### 1.2. Functional Requirements:

* Accuracy and efficiency.
* Flexibility.
* Availability and reliability.
* Security and privacy.
* Usability.

### 1.3. Hardware Requirements:

* PC with efficient computational power (CPU-GPU) and adequate memory.

### 1.4. Software Requirements:

* SQL Server Management System.
* SQL server
* SQL server data tools
* Visual studio
* SQL server Reporting server
* Power bi Desktop application
* Draw.io application

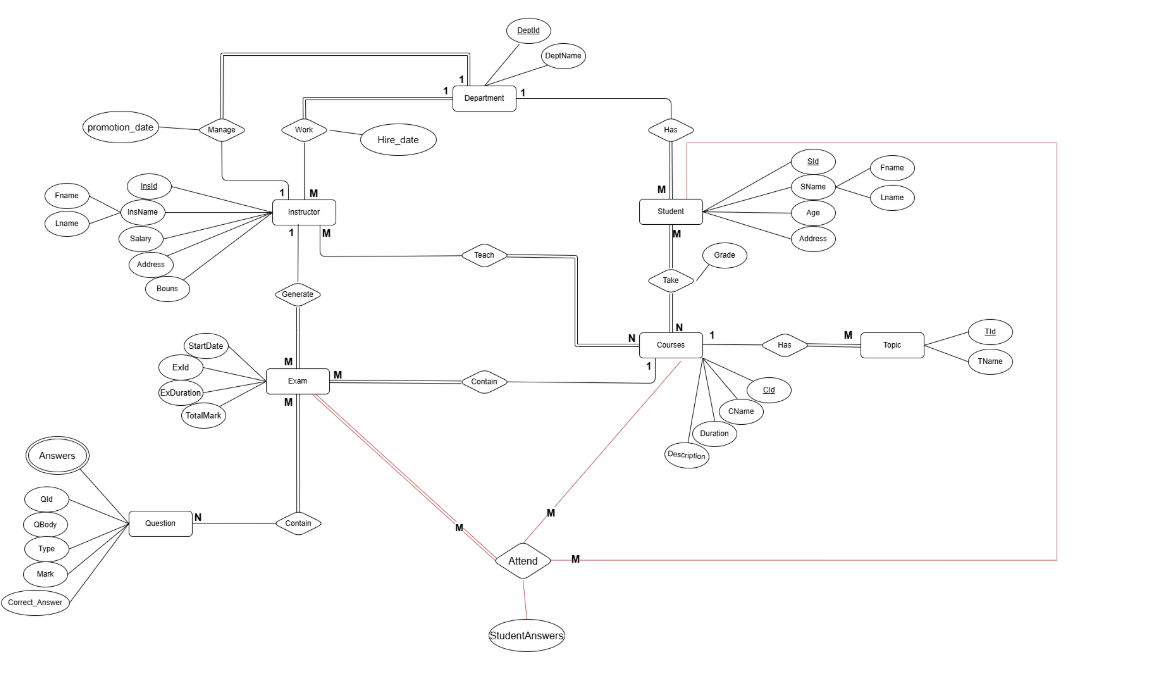
## Project Diagrams

* 1. ERD

Figure 1: ERD Screenshot



Pdf 2 ERD File Link



* 1. Mapping

A screenshot of a computer

Description automatically generated

Figure 3: Mapping diagram.



Pdf 4: Mapping diagram file link.

# Chapter Three: implementation:

1. **Database Design and Table Mapping:** The initial step involved designing the database schema by mapping out the relationships between different entities in the system, such as students, courses, exams, and results. We ensured that each entity was properly represented with appropriate attributes and data types, laying the foundation for a robust and scalable system.
2. **Database Creation:** Following the design phase, we created the main database, named **Examination\_System**, on Microsoft SQL Server, ensuring that the structure supports all functional and non-functional requirements of the system. The database was optimized for performance and scalability.
3. **Security Implementation:** We implemented SQL Server authentication mode, following best practices for security. This approach ensures that user credentials are securely managed and stored, minimizing the risk of vulnerabilities that could be exploited by attackers. By using SQL Server authentication, we ensure that only authorized users can access the system.
4. **User Management:** To facilitate collaboration among team members, we generated user accounts with specific logins for each member of the development team. This enabled efficient access control and ensured that each team member had the necessary privileges for their tasks.
5. **Table Creation for Entities:** We began building tables for each entity related to the system, such as **Students**, **Instructors**, **Courses**, **Departments**, **Exams**, **Results**, and **Questions**. Each table was designed to adhere to normalization standards, ensuring data integrity and reducing redundancy.
6. **Stored Procedures for Data Manipulation:** We developed stored procedures to handle **DML (Data Manipulation Language)** operations, including Insert, Update, and Delete operations, as well as Select queries to interact with the tables. These stored procedures ensure optimized performance and maintain consistency across data operations.
7. **Exam Generation and Correction:** The next step involved creating stored procedures and logic to handle exam generation, exam corrections, and the creation of model answers for student exams. These functionalities are crucial for automating exam preparation and ensuring consistency in assessment.

A screenshot of a computer program

Description automatically generated

Figure 5;Exam Generation SP part1

**A screenshot of a computer code

Description automatically generated**

Figure 6:Exam Generation SP part 2

A screenshot of a computer program

Description automatically generated

Figure 7:Student Model Answers SP

A screenshot of a computer

Description automatically generated

Figure 8: Exam Correction SP Part1

A screenshot of a computer

Description automatically generated

Figure 9:Exam correction SP part2

1. **Triggers for Business Logic and Data Integrity:** To enforce business rules and constraints, we created **triggers** as a type of stored procedure. These triggers help maintain data integrity by restricting unauthorized actions and ensuring that all operations on the database adhere to the defined business logic. Additionally, we implemented an **auditing table** to track all significant actions performed within the system, including the action type, timestamp, and the user who executed it, thus ensuring transparency, integrity, and accountability.
2. **Reporting and SSRS Integration:** We developed stored procedures for generating required reports, including exam results, student performance, and attendance summaries. These reports were then designed and deployed using **SQL Server Reporting Services (SSRS)**, **Visual Studio**, and **SQL Server Data Tools (SSDT)**. This integration enables dynamic report generation and makes it easy to access and analyze data.

A screenshot of a computer code

Description automatically generated

Figure 10:1st Report SP

A screen shot of a computer code

Description automatically generated

Figure 11:2nd Report SP

A screenshot of a computer program

Description automatically generated

Figure 12:3rd Report SP

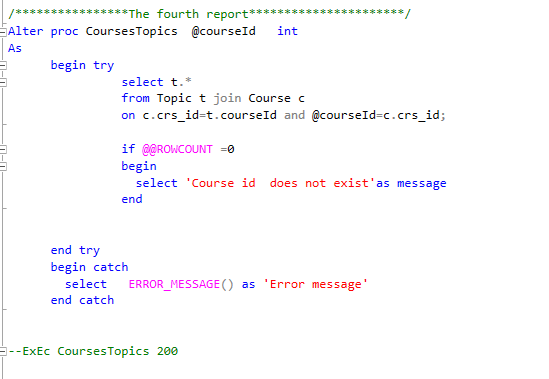


Figure 13:4rt Report SP

A screenshot of a computer code

Description automatically generated

Figure 14:5th Report SP

A screenshot of a computer program

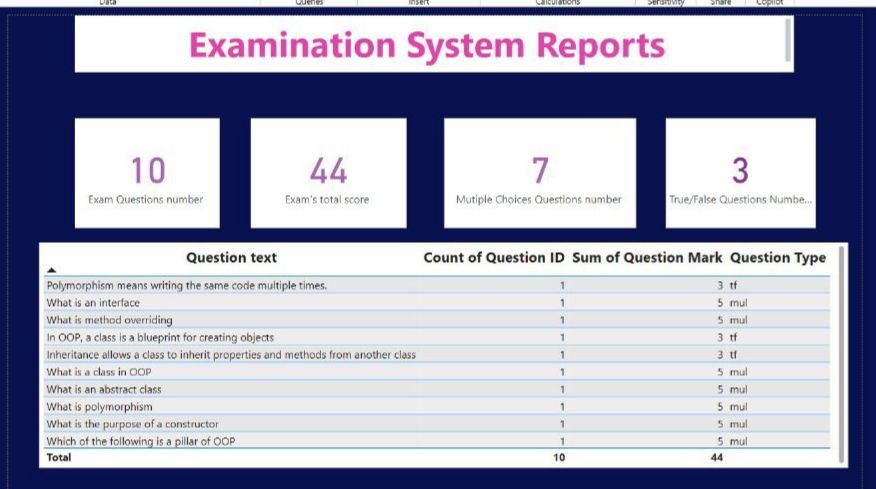
Description automatically generated

Figure 15:6th Report SP





Then, we tried to construct a Power BI dashboard for these 6 reports. The following screenshot illustrates it.



A screenshot of a computer screen

Description automatically generatedFigure 16: Exam Question Report dashboard

Figure 17: Student by department and student grades Report dashboard

A screenshot of a computer

Description automatically generated

Figure 18Courses Topics Report dashboard

1. **Data Population and Performance Optimization:** After fulfilling the core functionality requirements, we populated the database with initial data for students, instructors, courses, topics, departments, exams, and questions. Rather than manually executing SQL queries each time, we leveraged the stored procedures we had previously created to automate data insertion and ensure fast, efficient performance. This approach significantly reduced execution time, improved system responsiveness, and enhanced overall efficiency.