**Yarmouk Private University**

**Faculty of Informatics and Communication Engineering**

**Department Software Engineering**


# Education Assessment System

A project to gain the Bachelor degree in software engineering

**Developed By**

Seif Al-Deen Al-Kowatli Salim Jammoul

**Supervised By**

Dr. Eng. Tala Shehbi

Eng. Jihad Allala

Semester

2017/2018-2

# ABSTRACT

In this project we developed an Education Assessment System (EAS). EAS is a distributed system consist of two main API servers; Examination System API and Assessment and Analysis API, and two ASP.MVC end points; Examination endpoint and University Portal Endpoint. EAS use the Multiple-Choice Questions (MCQ) approach to perform assessment process. And provide three types of assessment; placement, summative and formative assessment. EAS uses the data from the examination process to perform assessment on the course, student and teacher, and return results in the form of different types of graphs and histograms which reflect the status of the course, student and teacher and help spotting problem, to improve the education process.

# Table of Contents

# ACKNOWLEGMENT

First, we would like to give special thanks to our supervisor Dr. Wassim Juneidi for his exceptional direction and expertise as well as his encouragement to maintain our progress in track. Dr. Wassim Juneidi helped us going in some of toughest times during both junior and senior projects. This work could not have been completed without his unconditional support and motivation.

We would also like to express our special thanks to our supervisor Dr. Talal Shehabi for his continuous support in our senior project, who have given his full effort in guiding us in achieving the goal. He taught us how to ask questions and express our ideas. We have been extremely lucky to have a supervisor who cared so much about our work.

Last, but not least, we would like to express our gratitude and appreciation to all those who gave us the possibility to complete this whole journey. We would like to thank all Doctors and Professors who taught us and our family and friends who gave us support and believed in us.

# 1st Chapter

## (Introduction)

# 1. Aim

Educational assessment is the systematic process of documenting and using empirical data on the knowledge, skill, attitudes, and beliefs to refine programs and improve student learning. Assessment data can be obtained from directly examining student work to assess the achievement of learning outcomes or can be based on data from which one can make inferences about learning. Assessment is often used interchangeably with test, but not limited to tests. Assessment can focus on the individual learner, the learning community (class, workshop, or other organized group of learners), a course, an academic program, the institution, or the educational system as a whole (also known as granularity). The word 'assessment' came into use in an educational context after the Second World War.

As a continuous process, assessment establishes measurable and clear student learning outcomes for learning, provisioning a sufficient amount of learning opportunities to achieve these outcomes, implementing a systematic way of gathering, analyzing and interpreting evidence to determine how well student learning matches expectations, and using the collected information to inform improvement in student learning.

The final purpose of assessment practices in education depends on the theoretical framework of the practitioners and researchers, their assumptions and beliefs about the nature of human mind, the origin of knowledge, and the process of learning.

Assessment is often divided into initial, formative, and summative categories for the purpose of considering different objectives for assessment practices

Now days technology spreads over everything including the educational field. Exams are computer based same for the grading and assessment. Because

computer-based examination and grading guarantee fairness and eliminate human bias, and computer-based assessment gives more reliable and more accurate results.

That's why we built our Education Assessment System **(EAS).** Our EAS start the assessment process from the stage of examination by guaranteeing that exam have no bias and all students have equal chance to fail or succeed, which eliminates the bias during the assessment process and reflects more accurate results. EAS apply advanced algorithms on the examination data and gives results which reflect the status of the students, teachers and curriculum. To build such system we had to not just built the assessment system, but also, we rebuilt our previous examination system and apply more advance algorithms to ensure that our examination process is reliable and academic.

# 2. THE SCOPE OF THE WORK

As we said EAS is take in consideration the accuracy of the examination data and ensures that examination process is academic and reliable to make sure that assessment is accurate so that our system is not consist of multiple subsystems.

We have two cores of EAS Examination and Assessment. Each one of them have its own API and provide multiple services. The examination core handles the examination process. Starting with creating the exam and generating the exam questions pattern to correction and grading. While the Assessment core handles assessment and analysis.

Those cores provide these services to the ASP.MVC Endpoints which request the services present it to the end user in a user-friendly look.

Since the information we work with are Examination Questions and Students data which is confidential, we put a lot of effort into the security part of the project and used encryption to ensure that the data is safe.

Our system examination and assessment algorithms can used to perform two types of assessment; summative, placement and formative assessment. In our University case study our main focus was the summative assessment.

# 3. DEEFINITIONS AND ABBREVIATIONS

This section contains the definition of all used terms, acronyms and abbreviations in this report.

*Table 1:Definitions*

| Definitions | |
|---|---|
| EAS | Our project name and it is an abbreviation of (Education Assessment System). |
| Application | A computer program designed to perform a group of coordinated functions for the benefits of the user. |
| User | Our main actor. |
| Admin | Actor with privilege. |
| Role | The action and the activities assigned to or required. |
| Platform | Whatever a pre-consisting piece of computer software is designed to run within, obeying its constraints, making use of the its facilities. |
| Algorithm | A self-contained step-by-step set of operations to be performed. |
| Plugin | A software component that adds a specific feature to an existing computer program. |
| Widget | An easy to use software application or component made for one or more different software platform. |

*Table 2:Abbreviations*

| Abbreviations | |
|---|---|
| EAS | Education Assessment System |
| OES | Online Examination Subsystem |
| ESA | Software Engineering Standards (ESA PSS-50-0 Issue 2) |
| UML | Unified Modeling Language |
| ERD | Entity Relationship Model |
| MVC | Model Viewer Controller |
| API | Application Programing Interface |
| LINQ | Language Integrated Query |
| APP | Application |
| SQL | Structured Query Language |
| HTML | Hypertext Markup Language |
| CSS | Cascading Style Sheet |
| XML | extensible markup language |
| Ajax | Asynchronous JavaScript and XML |
| REST | Representational State Transfer |
| WCF | Windows Communication Foundation |
| GCD | **greatest common divisor** |
| MCQ | Multiple-Choice Questions |

# 4. OVERVIEW

This document is structured according to the described in (ESA). In the second chapter you can find the different approaches of the process of education assessment and way MCQ approaches is the best fit with our system. Also, the types of assessments which can be provided by our system and used environment to implement the system, along with a deep description of the used techniques and technologies. Chapter three contains full details about the process of collecting and analyzing system requirements all the way to get to the stage of design facility with UML charts and tables required in addition to display the process of testing that applied system.

The fourth chapter discuss system algorithms and libraries and tools used in the development process. We discussed two types of algorithms OES algorithms and assessment algorithms.

Finally, you'll our future plans and how we can improve our system and how to integrate our system with the current education system along with all the preferences used to accomplish this work.

# 2nd Chapter

## (General Description)

# 1. Assessment Approaches

## 1.1. Essays and Text-based assignments

**Learning outcomes:**

Encourages the development of academic writing skills, and skills in extended argument, inquiry-based learning. Assesses relational and extended abstract SOLO level.

**Issues to consider:**

Time consuming to assess, need to develop practices to ensure consistency in grading. Plagiarism may be an issue.

## 1.2. Group Work

**Learning outcomes:**

Can assess process and product. Opportunities to develop skills in collaboration and co-operation. An example of authentic learning and assessment. Assesses relational and extended abstract SOLO level. Issues to consider: May be difficult to assess individual contribution. Peer pressure may be a problem. Can be difficult for students to meet. Students may not collaborate or cooperate, so time needs to be invested in laying down ground rules for this and assessment criteria and processes for individual participation/contribution.

## 1.3. Multiple-choice questions (MCQ)

**Learning outcomes:**

Useful for diagnostic, formative and summative assessment. Opportunity to provide feedback. Once established have minimal administration, suitable for large classes. Can be effective for assessing unistructural and multiscriptual SOLO levels.

**Issues to consider:**

Appropriate MCQs are difficult to design. May only encourage assessment at unistructural and multiscriptual SOLO levels.

# 2. Environments

## 2.1. Entity Framework

Entity Framework is data access technology for applications in .NET. It is an object relational mapper that enables .NET developers to work with relational data using domain specific objects. It eliminates the need for the most of the data-access code that developers usually need to write.

### How Entity Framework works?

The Entity Framework includes the EntityClient data provider. This provider manages connections, translates entity queries into data source-specific queries, and returns a data reader that the Entity Framework uses to materialize entity data into objects. When object materialization is not required, the EntityClient provider can also be used like a standard ADO.NET data provided by enabling applications to execute Entity SQL queries and consume the returned read-only data reader.

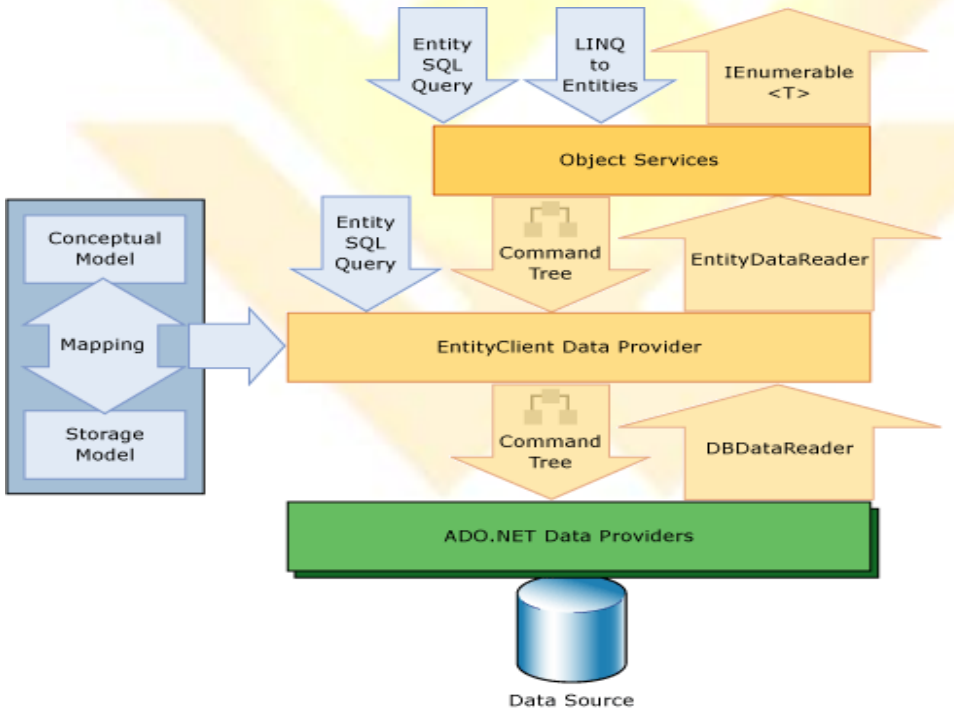



*Figure 1:Architecture of Entity Framework*

## Entity Framework

Entity Framework is an ORM and ORMs are aimed to increase the developer's productivity by reducing the redundant task of persisting the data used in the applications.

- Entity Framework can generate the necessary database commands for reading or writing data in the database and execute them for you.

- If you're querying, you can express your queries against your domain objects using LINQ to entities.

- Entity Framework will execute the relevant query in the database and then materialize results into instances of your domain objects for you to work within your app.

There are other ORMs in the marketplace such as NHibernate and LLBLGen Pro. Most ORMs typically map domain types directly to the database schema.

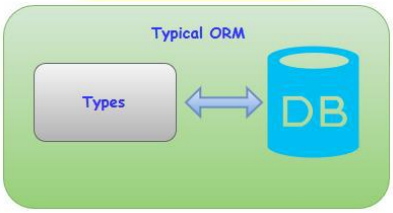



*Figure 2: ORM in Entity Framework*

Entity Framework has a more granular mapping layer so you can customize mappings, for example, by mapping the single entity to multiple database tables or even multiple entities to a single table.

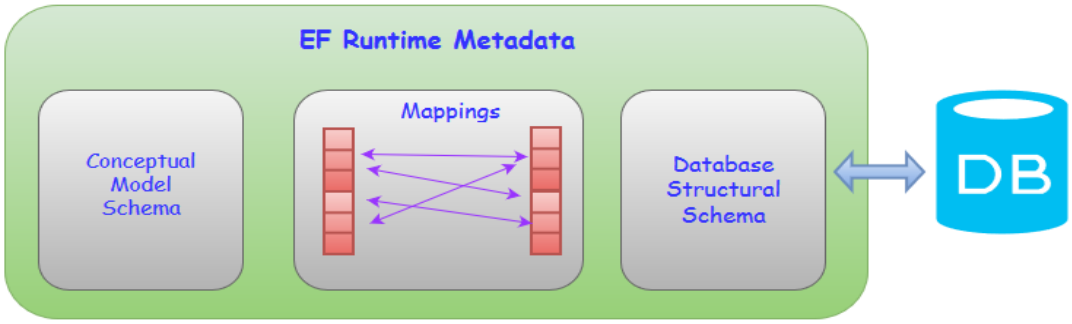



*Figure 3:The Mechanism of Entity Framework*

## 2.2. ASP.NET Web API

A **Web API** is an application programming interface (API) for either a web server or a web browser, usually limited to a web application's client-side (including any web frameworks being used). Asp.Net Web API is a framework for building HTTP services that can be consumed by a broad range of clients including browsers, mobiles, iPhone and tablets. It is very similar to ASP.NET MVC since it contains the MVC features such as routing, controllers, action results, filter, model binders, IOC container or dependency injection. But it is not a part of the MVC Framework. It is a part of the core ASP.NET platform and can be used with MVC and other types of Web applications like Asp.Net Web Forms. It can also be used as a stand-alone Web services application.

### Why Asp.Net Web API (Web API)?

Today, a web-based application is not enough to reach its customers. People are very smart, they are using iPhone, mobile, tablets etc. devices in its daily life. These devices also have a lot of apps for making the life easy. Actually, we are moving from the web towards apps world.

So, if you like to expose your service data to the browsers and as well as all these modern devices apps in fast and simple way, you should have an API which is compatible with browsers and all these devices.

For example, twitter, Facebook and Google API for the web application and phone apps.

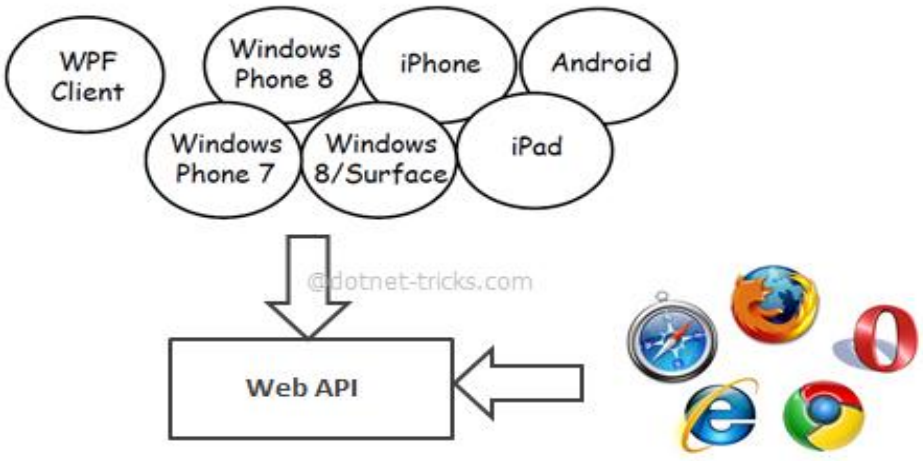



*Figure 4:How API Connects Everything*

**Web API Features**

1. It supports convention-based CRUD Actions since it works with HTTP verbs GET, POST, PUT and DELETE.
2. Responses have an Accept header and HTTP status code.
3. Responses are formatted by Web API's Media Type Formatter into JSON, XML or whatever format you want to add as a Media Type Formatter.
4. It may accept and generates the content which may not be object oriented like images, PDF files etc.
5. It has automatic support for OData. Hence by placing the new [Query able] attribute on a controller method that returns IQueryable, clients can use the method for OData query composition.
6. It can be hosted with in the application or on IIS.
7. It also supports the MVC features such as routing, controllers, action results, filter, model binders, IOC container or dependency injection that makes it more simple and robust.

**Web API is the best option for our Online Exam System**:

for exposing your data and service to different devices. Moreover, Web API is open source an ideal platform for building REST-full services over the .NET Framework. Unlike WCF Rest service, it uses the full features of HTTP (like URIs, request/response headers, caching, versioning, various content formats) and you don't need to define any extra configuration settings for different devices.

**unlike WCF Rest service:**

APIs available on both the client side and the server side:

makes Web programming easier allowing programmers to build web application on top of a high-level interface.

## ASP.NET Web VS. WCF:

WCF is Microsoft's unified programming model for building service-oriented applications. It enables developers to build secure, reliable, transacted solutions that integrate across platforms and interoperate with existing investments

The following table describes the major features of each technology:

*Table 3: WCF vs Web API*

| WCF | ASP.NET Web API |
|---|---|
| Enables building services that support multiple transport protocols (HTTP, TCP, UDP, and custom transports) and allows switching between them. | HTTP only. First-class programming model for HTTP. More suitable for access from various browsers, mobile devices etc enabling wide reach. |
| Enables building services that support multiple encodings (Text, MTOM, and Binary) of the same message type and allows switching between them. | Enables building Web APIs that support wide variety of media types including XML, JSON etc. |
| Supports building services with WS-* standards like Reliable Messaging, Transactions, Message Security. | Uses basic protocol and formats such as HTTP, WebSockets, SSL, JQuery, JSON, and XML. There is no support for higher level protocols such as Reliable Messaging or Transactions. |
| Supports Request-Reply, One Way, and Duplex message exchange patterns. | HTTP is request/response but additional patterns can be supported through SignalRand WebSockets integration. |
| WCF SOAP services can be described in WSDL allowing automated tools to generate client proxies even for services with complex schemas. | There is a variety of ways to describe a Web API ranging from auto-generated HTML help page describing snippets to structured metadata for OData integrated APIs. |
| Ships with the .NET framework. | Ships with .NET framework but is open-source and is also available out-of-band as independent download. |

## 2.3. ASP.NET MVC

### What is MVC?

- Model-View-Controller (MVC).
- Standard Architectural Pattern.
- Separation of concerns: model, view, controller.

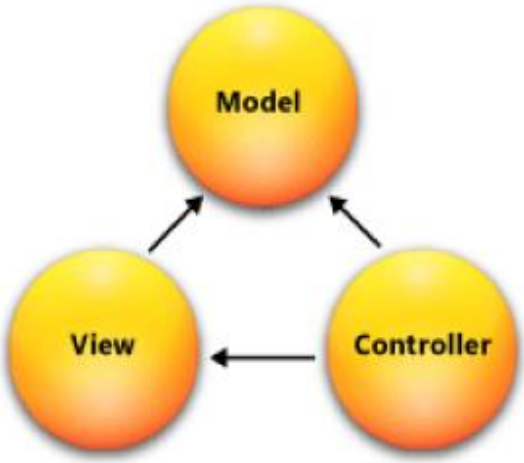



*Figure 5:MVC Architecture*

### ASP .NET MVC Framework Components

- **Models**
  - Business/domain logic.
  - Model objects, retrieve and store model state in a persistent storage.
- **Views**
  - Display application's UI.
  - UI created from the model data.
- Controllers
  - Handle user input and interaction.
  - Work with model.
  - Select a view for rendering UI.

There are two MVC projects in our system each one handles certain types of clients

I. **Online Examination:** handles the online examination process and send requests through the intranet
II. **Online Examination Gate & Admin Panel:** consist of three main sections
    1. **Students Gate:** Allow students to show their exams and requests detailed report about a certain exam.
    2. **Teacher Gate:** Allow Teacher to Add, Modify or Delete question of certain subject.
    3. **Admin Panel:** admin can manage accounts and create new exams.

## MVC Request

To fetch data from the web API server, we must use a technique can bring life into UI.

MVC Endpoint Able to fetch Data from the API server According to the following steps:

1. Build new MVC controller that return view of the wanted Interface.
2. Add new view .in this view we can use html with JavaScript to send and fetch data from the web API server by use jQuery $. ajax that send an XMLHttpRequest (XHR) object to the server. This XHR object (which is a part of Ajax) includes data that tells the server what is being requested. The API server then responds with only the data that was requested for. When the API server responds with the data, the browser uses JavaScript also to receive the data, processes it and updates only a portion of the page that has changed. All this occurs asynchronously in the background without any page reloads; while the user continues working on the other parts of your webpage. This gives the user a more responsive and natural experience. After the web browser sends off a request to the server using the XML Http Request object, it waits for a response from the API server. When the API server responds. Also, we use jQuery callbacks to handle callbacks for success and failure:
    a. done (response, status, jqXHR) - called when the response from the server is successful
    b. fail (jqXHR, status, error) - called when the response from the server fails or the request times out.
    c. complete (response, status, jqXHR) - Always called when a response is received from the server

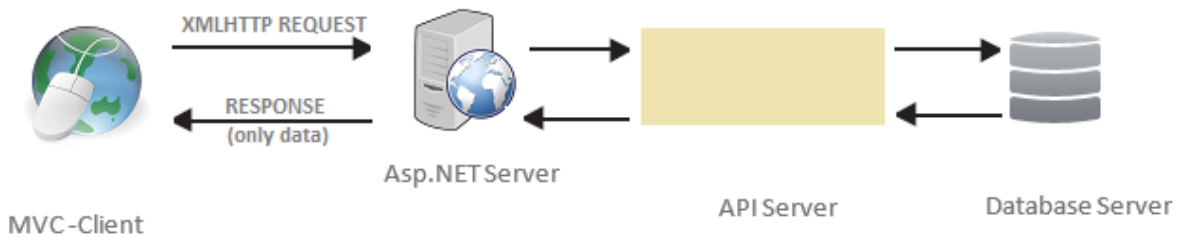



*Figure 6: How Ajax Request is Made*

**Technologies Used in MVC to call the web API:**

We used the Web Standard Model: HTML & CSS to create our pages and we used the JavaScript to add Animation and to call the API. What is all those technologies.

**HTML & CSS:**

HTML (the Hypertext Markup Language) and CSS (Cascading Style Sheets) are two of the core technologies for building Web pages. HTML provides the structure of the page, CSS the (visual and aural) layout, for a variety of devices.

**JavaScript:**

a high-level, dynamic, untyped, object-based, multi-paradigm, and interpreted programming language. Alongside HTML and CSS, JavaScript is one of the three core technologies of World Wide Web content production. It is used to make webpages interactive and provide online programs, including video games. The majority of websites employ it, and all modern web browsers support it without the need for plug-ins by means of a built-in JavaScript engine. The two features of JavaScript that we used in our MVC projects are.

**jQuery:**

jQuery is a fast and concise JavaScript Library created by John Resig in 2006 with a nice motto: Write less, do more. jQuery simplifies HTML document traversing, event handling, animating, and Ajax interactions for rapid web development. jQuery is a JavaScript toolkit designed to simplify various tasks by writing less code. Here is the list of important core features supported by jQuery that we used in our system:

- **Event handling:** The jQuery offers an elegant way to capture a wide variety of events, such as a user clicking on a link, without the need to clutter the HTML code itself with event handlers.
- **AJAX Support:** jQuery support AJAX technology which helped us to get updates of the Website without the need to refresh the page.
- **Animations:** The jQuery comes with plenty of built-in animation effects which you can use in your websites.

**Ajax:**

Ajax stands for Asynchronous JavaScript and XML. AJAX is a new technique for creating better, faster, and more interactive web applications with the help of XML, HTML, CSS, and Java Script.

- Ajax uses XHTML for content, CSS for presentation, along with Document Object Model and JavaScript for dynamic content display.
- Conventional web applications transmit information to and from the sever using synchronous requests. It means you fill out a form, hit submit, and get directed to a new page with new information from the server.
- With AJAX, when you hit submit, JavaScript will make a request to the server, interpret the results, and update the current screen. In the purest sense, the user would never know that anything was even transmitted to the server.
- AJAX is a web browser technology independent of web server software.
- A user can continue to use the application while the client program requests information from the server in the background. Intuitive and natural user interaction. Clicking is not required, mouse movement is a sufficient event trigger.
- AJAX is based on the following open standards:
    - Browser-based presentation using HTML and Cascading Style Sheets (CSS).
    - Data is stored in XML format and fetched from the server.
    - Behind-the-scenes data fetches using XMLHttpRequest objects in the browser.
    - JavaScript to make everything happen.

# 3rd Chapter

**(Requirements & Design)**

# 1. Requirement Analysis

## 1.1. Functional Requirements

This section begins to describe in more specific and precise details exactly what steps the system takes in the course of its performance. Use case diagram serve not only to more specifically define system (and its boundaries), but also to identify functional requirements, to identify initial objects/classes, and to organize the work.

### 1.1.1. Use-Case Diagram

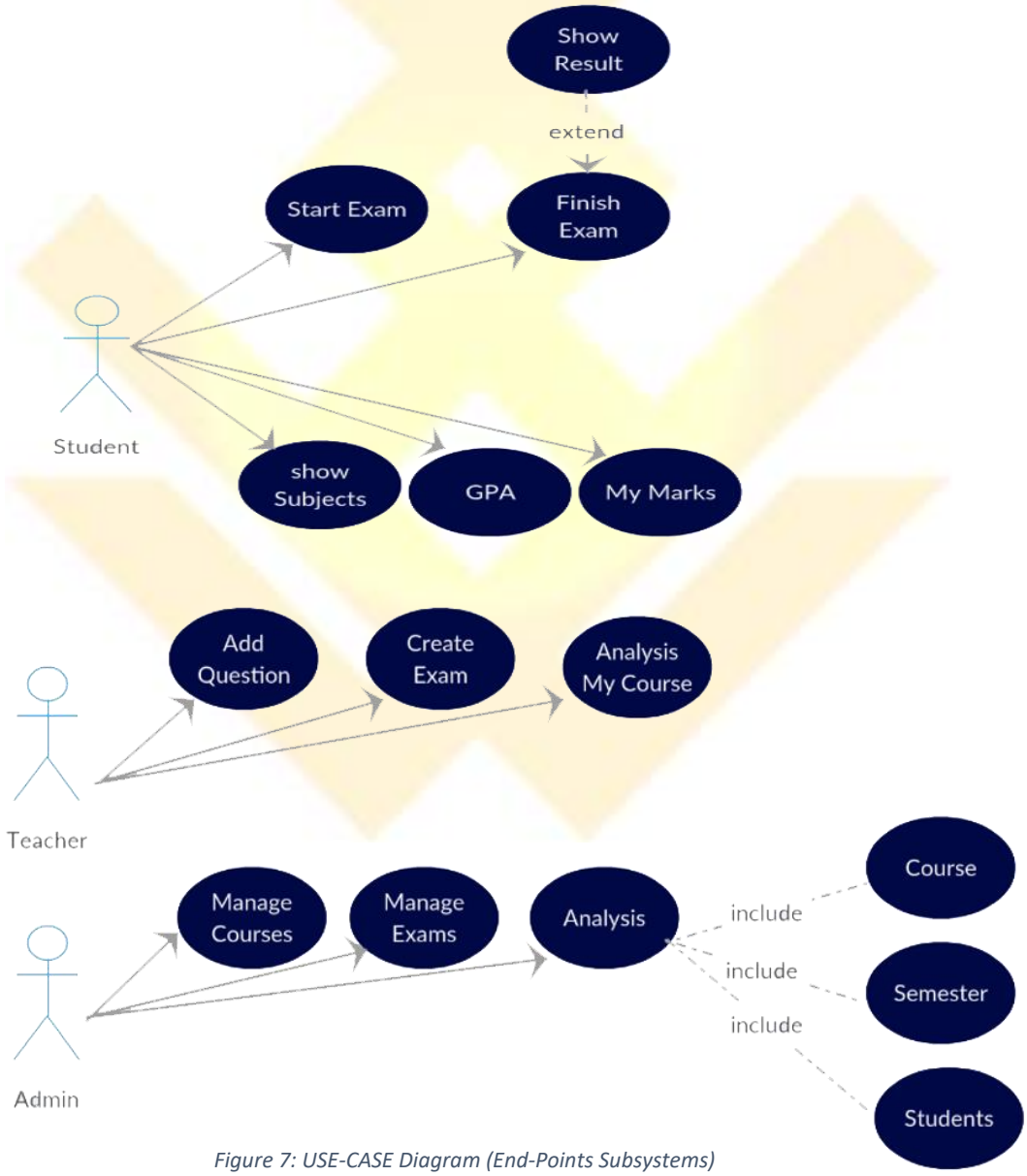



*Figure 7: USE-CASE Diagram (End-Points Subsystems)*

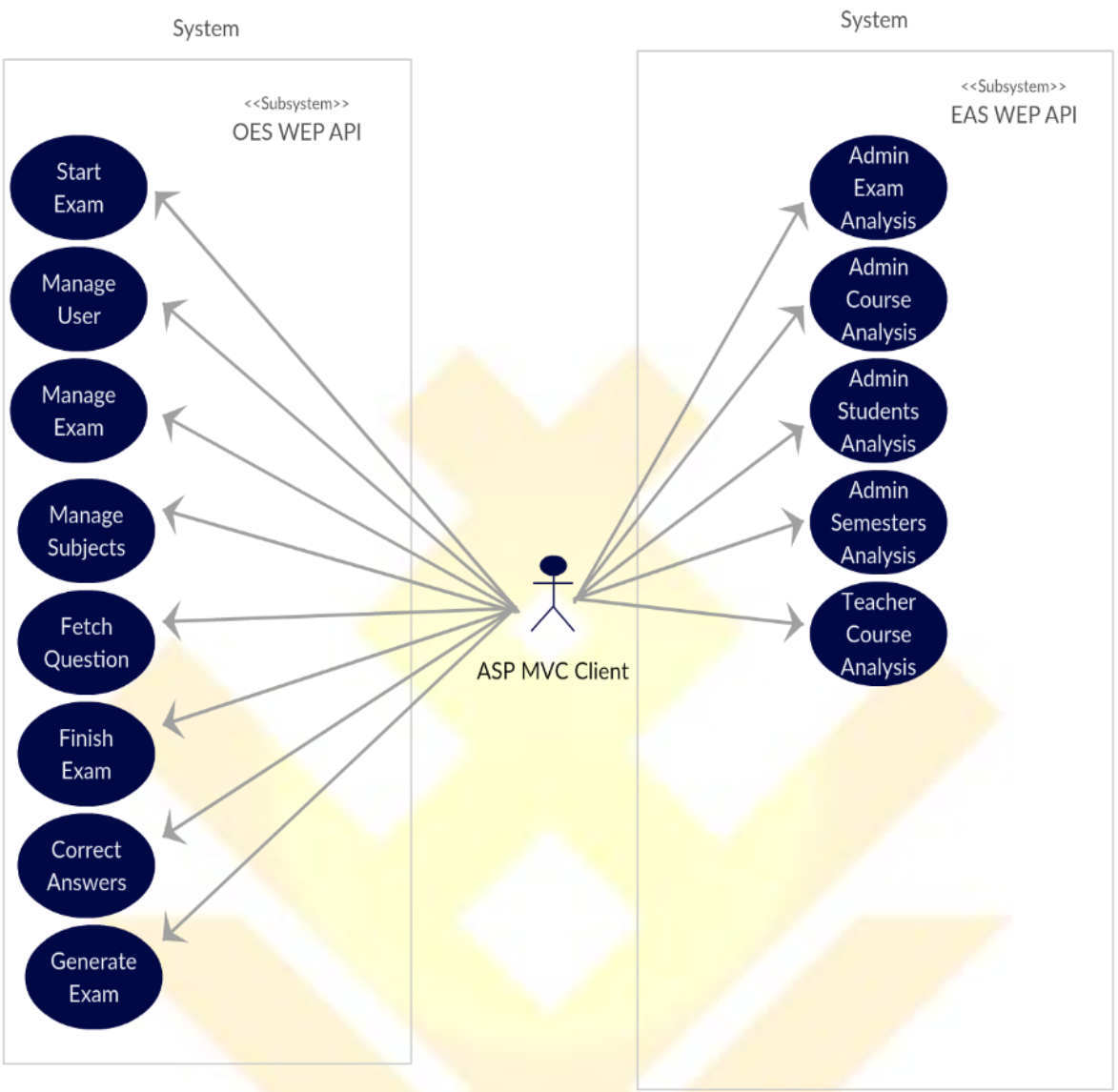



*Figure 8:USE-CASE Diagram (API Subsystems)*

## 1.1.2. Use-Case Specifications

In this section we will present our use case specifications in a detailed manner for each of our subsystems. We have four subsystems.

## A. Examination Endpoint

*Table 4:UC-A-1 Start the Exam*

| UC-A-1 | Start the Exam | |
|---|---|---|
| **Actor** | Student | |
| **Brief Description** | This use case describes how the student can start his exam. | |
| **Pre-Conditions** | • There is an active network connection to local server.<br>• Signed in as student. | |
| **Post-conditions** | • Start displays exam time.<br>• Display the first question. | |
| **Flow of events** | **Actor** | **System** |
| | 1. Click button “Start the Exam”. | |
| | | 2.Request for new exam and get first question. |
| | | 3.Display the first question with his Expected answers. |
| | | 4.Show the total time of the exam and begins to decrease it. |

| | | |
|---|---|---|
| | | 5. begins to decrease the specified time of Current question. |
| **Critical Scenarios** | • Error with login information<br>• Request to start the exam before the exam time so that the query algorithm is not yet started.<br>• It is forbidden to apply for the exam. | |

User Interface P

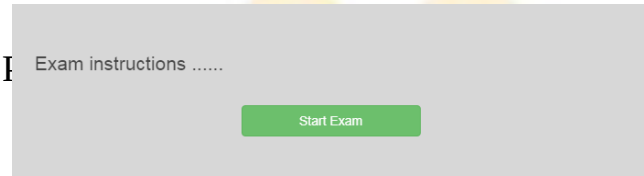



*Figure 9:Start Exam UI Prototype*

*Table 5:UC-A-2 Next Question*

| UC-A-2 | Next Question | |
|---|---|---|
| **Actor** | Student | |
| **Brief Description** | This use case describes how the student submit his answer to get next question. | |
| **Pre-Conditions** | • Signed in as student.<br>• Started the exam.<br>• Select his answer. | |
| **Post-conditions** | • Correct the student's answer and display the next question. | |
| **Flow of events** | **Actor** | **System** |
| | 1. Click button “submit”. | |
| | | 2.The system save student Answer |

| | | |
|---|---|---|
| | | 3.Requset for next question. |
| | | 4.Display the question. |
| **Critical Scenarios** | • Current question was the last one. | |

User interface prototype:

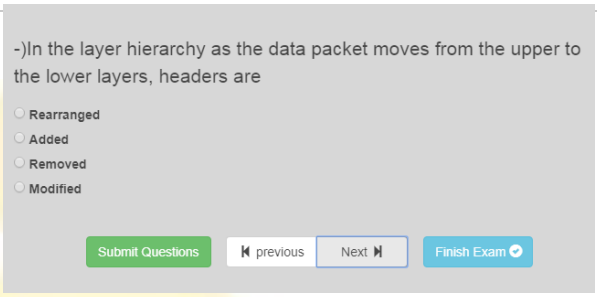



*Figure 10:Next Question UI Prototype*

*Table 6:UC-A-3 Previous Question*

| UC-A-3 | Previous Question | |
|---|---|---|
| **Actor** | Student | |
| **Brief Description** | This use case describes how the student request for previous question. | |
| **Pre-Conditions** | • Signed in as student.<br>• Started the exam. | |
| **Post-conditions** | • Get previous question with student answer's if he answered the question. | |
| **Flow of events** | **Actor** | **System** |
| | 1. Click button "previous". | |
| | | 2.The system save student Answer. |

| | | |
|---|---|---|
| | | 3.Requset for previous question. |
| | | 4.Display the question. |
| **Critical Scenarios** | • Current question was the first one. | |

*Table 7:UC-A-4 Submit the answer*

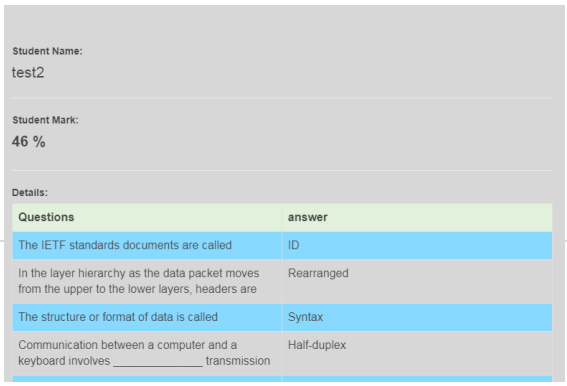
| **UC-A-4** | **Submit the answer** | |
|---|---|---|
| **Actor** | Student | |
| **Brief Description** | This use case describes how the student submit question with his answer. | |
| **Pre-Conditions** | • Signed in as student.<br>• Started the exam.<br>• Select his answer. | |
| **Post-conditions** | If student was sure his answer was correct ,he can submit the question then after submitted the question ,the question deletes from question list | |
| **Flow of events** | **Actor** | **System** |
| | 1. Click button “submit”. | |
| | | 2.The system get student answer and request for Correct it. |
| | | 3.Requset for next question. |
| | | 4.Display the question. |

| | |
|---|---|
| **Critical Scenarios** | • Submit unanswered question. |

*Table 8: UC-A-5 Finish the exam*

| **UC-A-5** | **Finish the exam** | |
|---|---|---|
| **Actor** | Student | |
| **Brief Description** | This use case describes how the student can finish his exam and get his final result. | |
| **Pre-Conditions** | • Sign in as student.<br>• Started the exam.<br>• Answer a number of specific questions. | |
| **Post-conditions** | Show the student final result. | |
| **Flow of events** | **Actor** | **System** |
| | 1. Click button “finish the Exam”. | |
| | | 2.The system confirm the request from student before execution. |
| | | 3.Requset for the final result of this student. |
| | | 4.Display the result with details. |
| **Critical Scenarios** | • None. | |

User Interface prototype:



# B. Portal Endpoint

*Table 9:UC-B-1 Sign in*

| UC-B-1 | Sign in | |
|---|---|---|
| **Actor** | Student, Admin, Teacher. | |
| **Brief Description** | This use case describes How to sign in.<br>redirect the user to their view that Available to them. | |
| **Pre-Conditions** | None. | |
| **Post-conditions** | • Redirect the user to view. | |
| **Flow of events** | **Actor** | **System** |
| | 1. Enter login information. | |
| | 2. Click Button “Sign in” | |
| | | 3. Request for view that Available to this user. |
| | | 4. Redirect to the view. |
| **Critical Scenarios** | • Error with login information | |

User Interface Prototype:


*Figure 11:Sign in UI Prototype*

*Table 10: UC-B-2*

| UC-B-2 | GPA | |
|---|---|---|
| **Actor** | Student. | |
| **Brief Description** | This use case describes how student can get his GPA. | |
| **Pre-Conditions** | • Logged in as Student. | |
| **Post-conditions** | • Show student rate in each semester than calculate student GPA | |
| **Flow of events** | **Actor** | **System** |
| | 1. Click" GPA" | |
| | | 2. Request for get student rate in each semester. |
| | | 3. Show student rate and his GPA. |
| **Critical Scenarios** | • Begin of semester | |

User interface prototype:

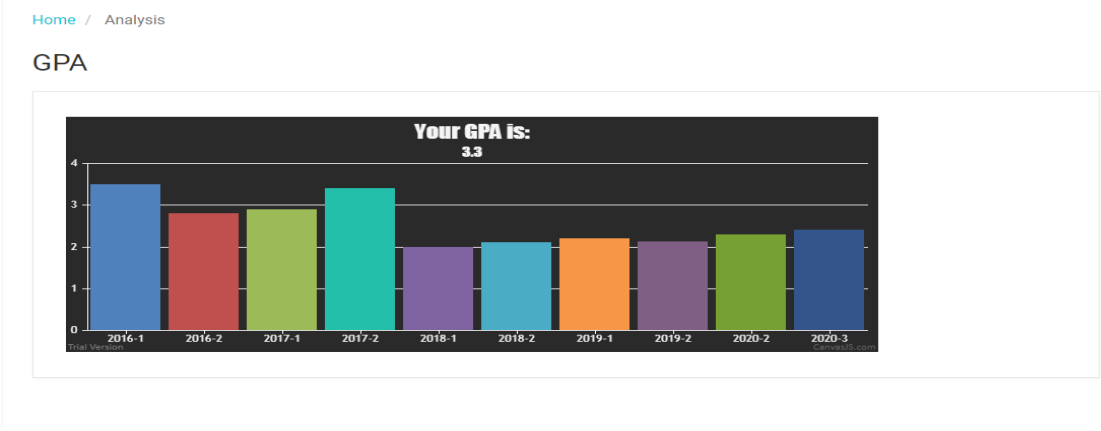



*Figure 12:GPA UI Prototype*

*Table 11:UC-B-3 Marks*

| UC-B-3 | Marks | |
|---|---|---|
| **Actor** | Student. | |
| **Brief Description** | This use case describes how student can show his marks in each course. | |
| **Pre-Conditions** | • Logged in as Student. | |
| **Post-conditions** | • Show student's marks in each course in the end of semester | |
| **Flow of events** | **Actor** | **System** |
| | 1. Click "Marks ". | |
| | | 2. Request for get student Marks in each Course. |
| | | 3. Show student Marks |
| **Critical Scenarios** | • Begin of semester | |

User i 2018-1

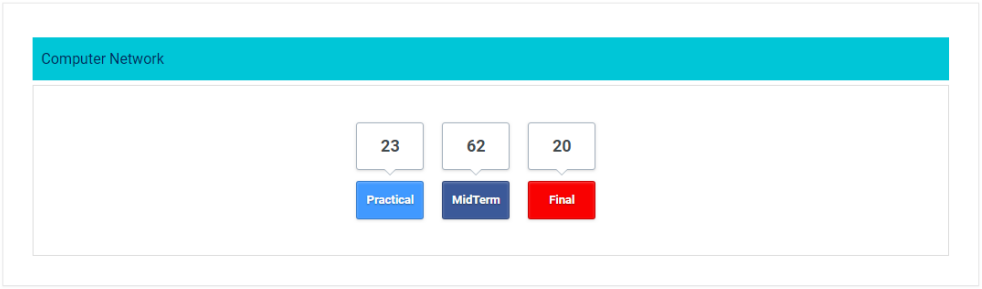



*Figure 13:Marks UI Prototype*

*Table 12:UC-B-4 Add Question*

| UC-B-4 | Add Question | |
|---|---|---|
| **Actor** | Teacher. | |
| **Brief Description** | This use case describes how Teacher can Add new Question. | |
| **Pre-Conditions** | • Logged in as Teacher. | |
| **Post-conditions** | • Teacher add new question in course. | |
| **Flow of events** | **Actor** | **System** |
| | 1. Click “Add Question”. | |
| | 2. Select course. | |
| | 3. Insert question text. | |
| | 4. Insert question 4 Answers. | |
| | 5. Insert question thinking skills. | |
| | 6. Insert question time | |
| | 7. Insert question difficulty. | |
| | 8. Insert question marks. | |
| | 9. Select topic. | |
| | 10. Click “Add”. | |

| | | |
|---|---|---|
| | | 11. Request for add new question. |
| | | 12. Show Result of the request. |
| **Critical Scenarios** | • Error in question information | |

User interface prototype:

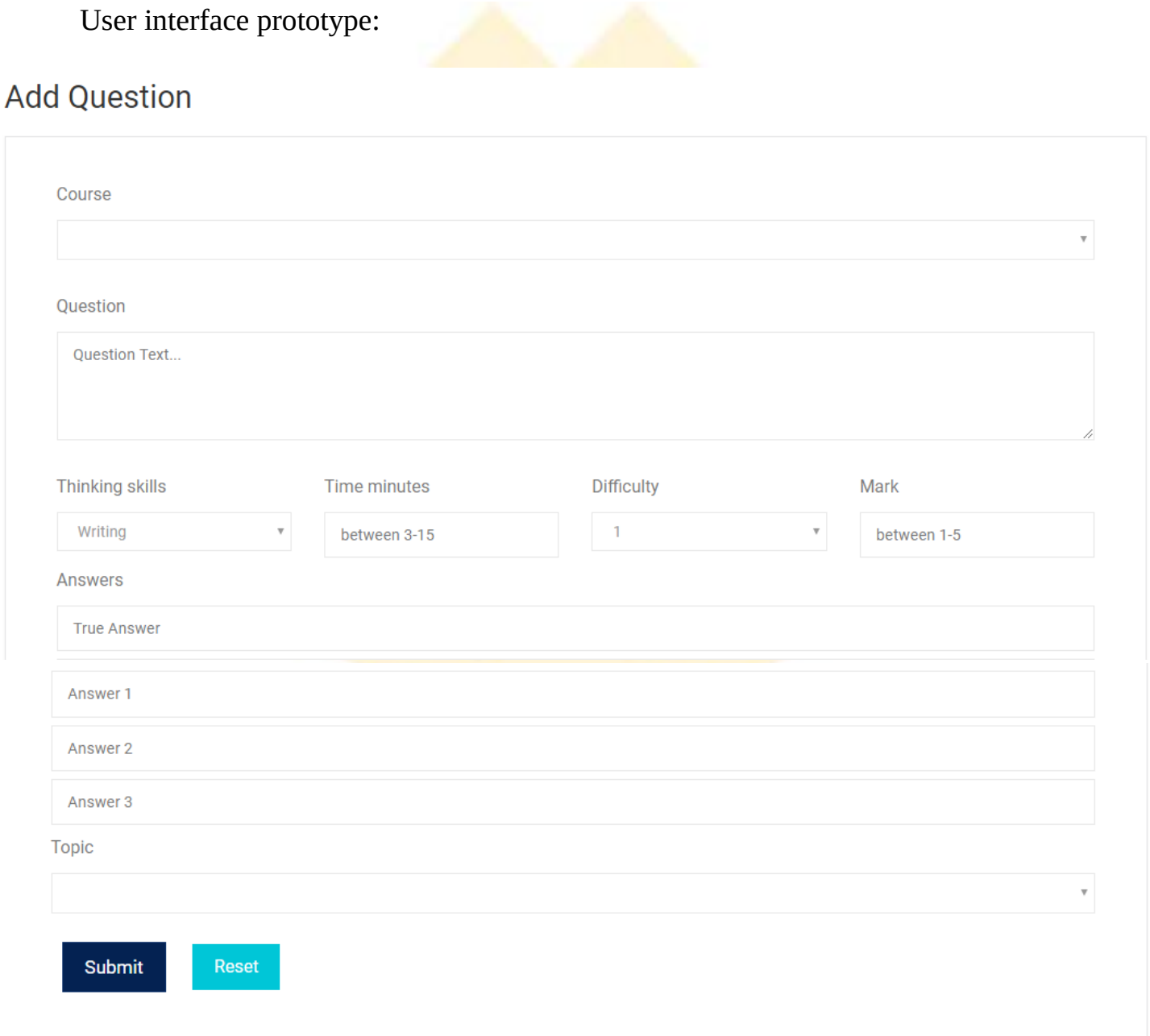



*Figure 14:Add Question UI Prototype*

*Table 13: UC-B-5 Create Exam*

| UC-B-5 | Create Exam | |
|---|---|---|
| **Actor** | Teacher. | |
| **Brief Description** | This use case describes how Teacher can create Exam. | |
| **Pre-Conditions** | • Logged in as Teacher. | |
| **Post-conditions** | • Teacher create exam and select exam topic and information | |
| **Flow of events** | **Actor** | **System** |
| | 1. Click “Create Exam”. | |
| | 2. Select course. | |
| | 3. Insert Exam total Grade. | |
| | 4. Insert Exam period. | |
| | 5. Select Exam topic. | |
| | 6. Select exam thinking skills. | |
| | 7. Select Exam difficulty. | |
| | 8. Click” finish”. | |
| | 9. Making sure the information is good. | |
| | 10. Click “Generate”. | |

| | | |
|---|---|---|
| | | 11. Request for create exam. |
| | | 12. Show “success”. |
| **Critical Scenarios** | • Error in Exam information | |

User interface prototype:

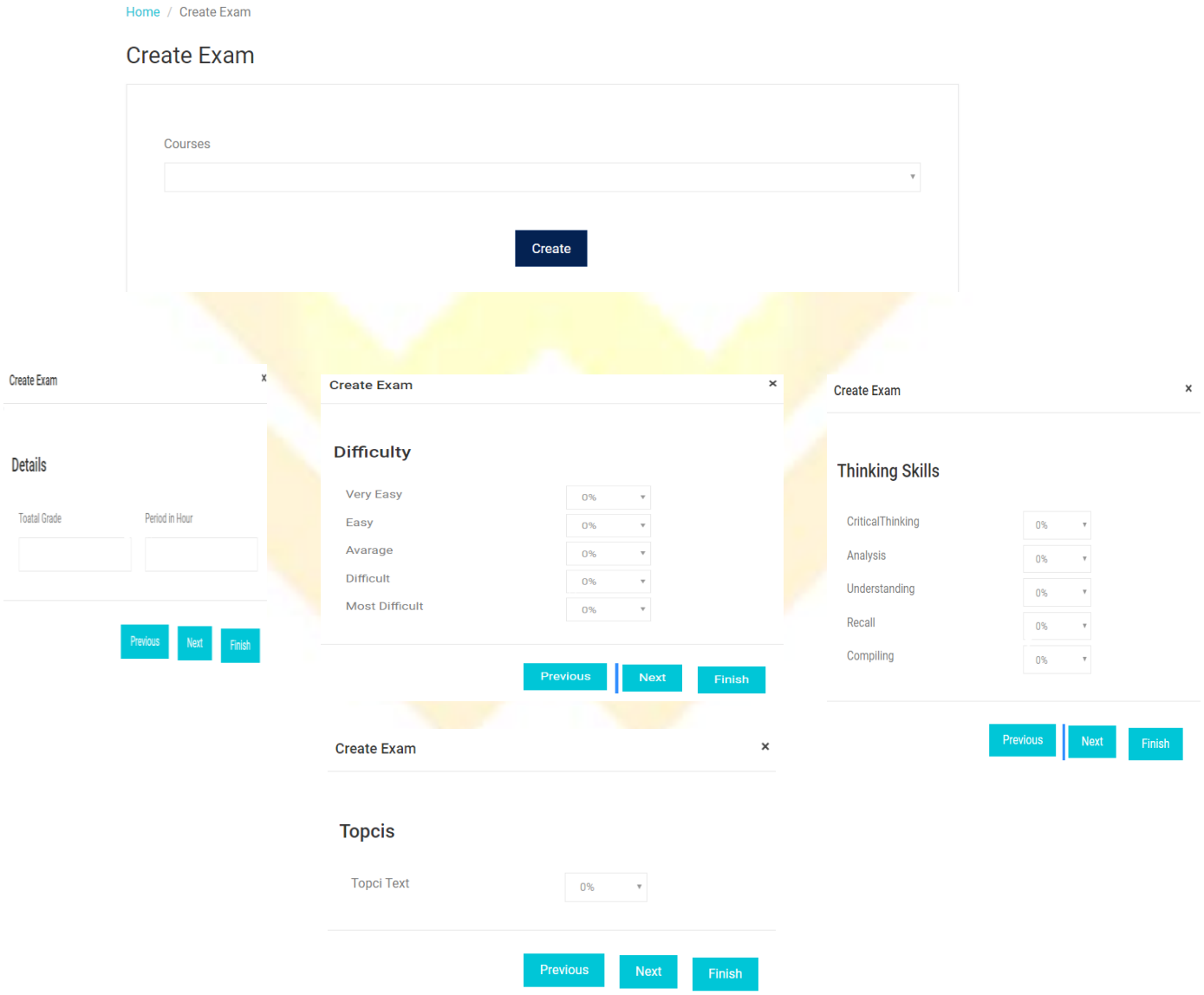



*Figure 15:Create Exam UI Prototypes*

*Table 14:UC-B-6 Course Analysis*

| UC-B-6 | Course Analysis | |
|---|---|---|
| **Actor** | Teacher. | |
| **Brief Description** | This use case describes how Teacher can get information about his courses | |
| **Pre-Conditions** | • Logged in as Teacher. | |
| **Post-conditions** | • Teacher get information about his courses like max mark, min mark and average mark. | |
| **Flow of events** | **Actor** | **System** |
| | 1. Click “Analysis” | |
| | 2. Select Semester | |
| | 3. Select Course | |
| | | 4. Request for get Course Information. |
| | | 5. Show chart about course information. |
| **Critical Scenarios** | • No information about course | |

User interface prototype:

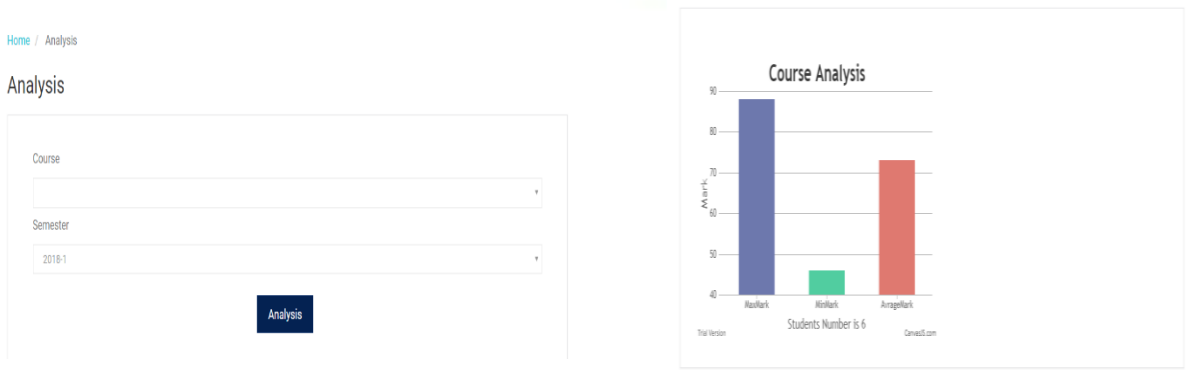



*Figure 16:Course Analysis UI Prototype*

| UC-B-7 | Add User | |
|---|---|---|
| **Actor** | Admin. | |
| **Brief Description** | This use case describes how Admin can add new user (Teacher or Student or Admin). | |
| **Pre-Conditions** | • Logged in as Admin. | |
| **Post-conditions** | • Add user information's, and send username and password in email to user, after that he can login the system | |
| **Flow of events** | **Actor** | **System** |
| | 1. Click "Add User" | |
| | 2. Add user information | |
| | 3. Click" Add". | |
| | | 4. Generate random password. |
| | | 5. Check the information and request for add new user. |
| | | 6.Show "success". |
| **Critical Scenarios** | • Error in user information | |

*Table 15:UC-B-7 Add User*

User interface prototype:


*Figure 17:Add User UI Prototype*

*Table 16:UC-B-8 Ad*

| UC-B-8 | Add New Course | |
|---|---|---|
| **Actor** | Admin. | |
| **Brief Description** | This use case describes how Admin can add new course with his ILOS. | |
| **Pre-Conditions** | • Logged in as Admin. | |
| **Post-conditions** | • Add new course to university courses with his Intended Learning Outcomes (ILOS). | |
| **Flow of events** | **Actor** | **System** |
| | 1. Click “Add Course” | |
| | 2. Add Course information and ILOS. | |
| | 3. Click” Add”. | |
| | | 4. Check the information and request for add new user. |

| | | |
|---|---|---|
| | | 5. Show request result |
| **Critical Scenarios** | • Error in Course information's. | |

User interface prototype:

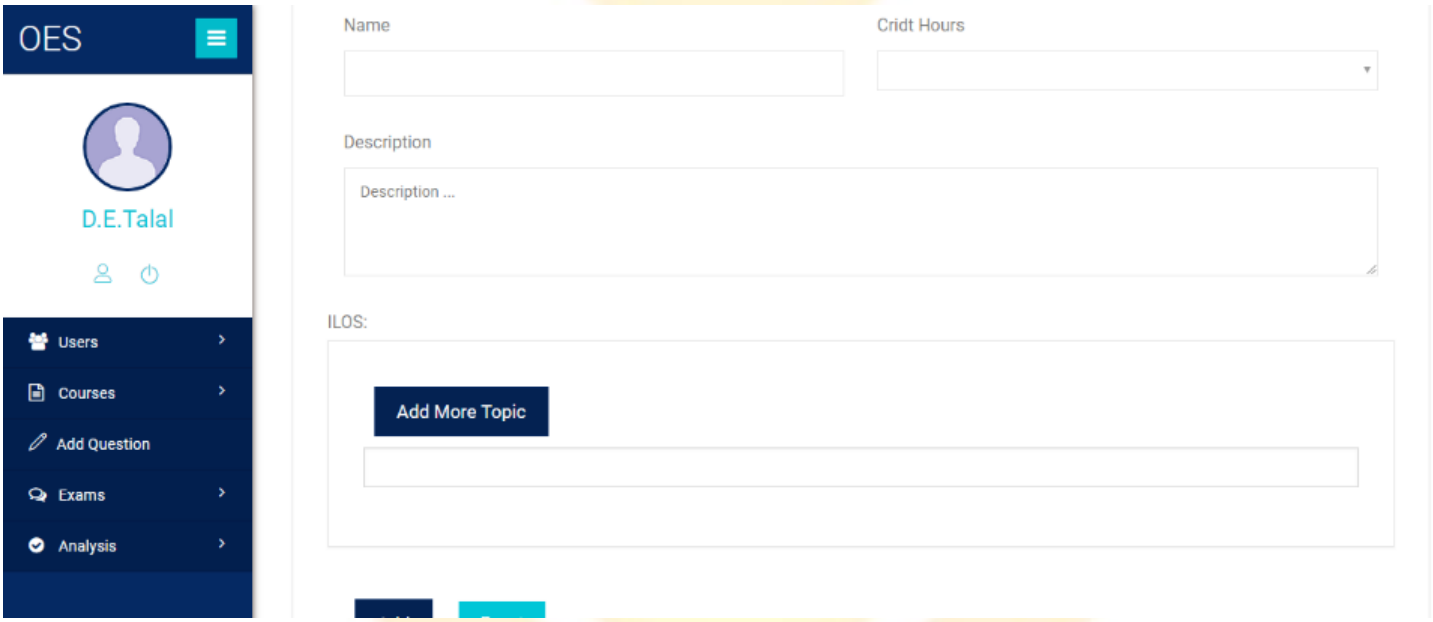
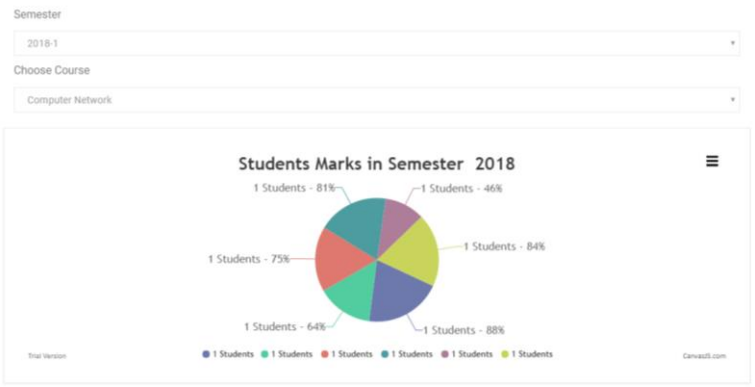



*Figure 18:Add New Course UI Prototype*

| UC-B-9 | Course statistics | |
|---|---|---|
| **Actor** | Admin. | |
| **Brief Description** | This use case describes how Admin can get students marks in course. | |
| **Pre-Conditions** | • Logged in as Admin. | |
| **Post-conditions** | • Get students marks and show the mark in chart | |
| **Flow of events** | **Actor** | **System** |
| | 1. Click “course statistics” | |
| | 2. Select Course | |
| | 3. Select semester | |
| | | 4. Check the information and request for get students mark |
| | | 5. Show “Student mark in chart” |
| **Critical Scenarios** | • No students mark. | |

*Table 17:UC-B-9 Course statistics*

User interface prototype:



| UC-B-10 | Add Exam | |
|---|---|---|
| **Actor** | Admin. *Figure 19:Course Statistics UI Prototype* | |
| **Brief Description** | This use case describes how Admin can add new Exam. | |
| **Pre-Conditions** | • Logged in as Admin. | |
| **Post-conditions** | • Add new exam. | |
| **Flow of events** | **Actor** | **System** |
| | 1. Click “Add Exam”. | |
| | 2. Select Course. | |
| | 3. Select Exam Type. | |
| | 4. Add Exam information | |
| | | 5. Check the information and request for get add new exam |
| | | 6. Show “Success” |
| **Critical Scenarios** | • Exam added by another Admin. | |

*Table 18:UC-B-10 Add Exam*

User interface prototype:

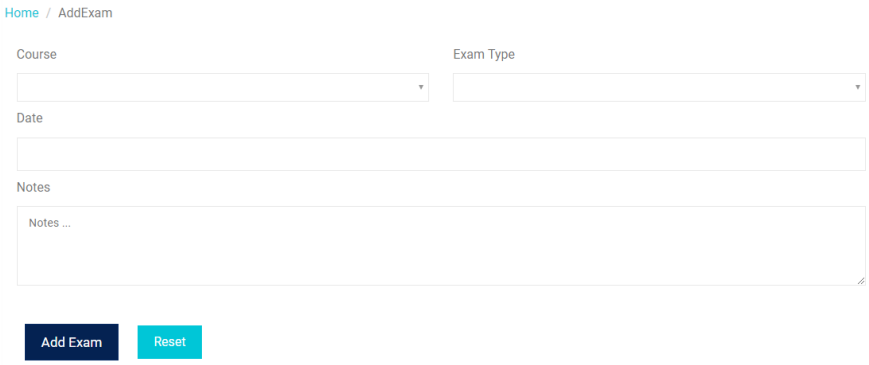



*Figure 20:Add Exam UI Prototype*

*Table 19:UC-B-11 Exam Analysis*

| UC-B-11 | Exam Analysis | |
|---|---|---|
| **Actor** | Admin. | |
| **Brief Description** | This use case describes how Admin can Analysis any Exam. | |
| **Pre-Conditions** | • Logged in as Admin. | |
| **Post-conditions** | • Get information about Exam like ILOS, Topic, success percentage. | |
| **Flow of events** | **Actor** | **System** |
| | 1. Click “Exam Analysis”. | |
| | 2. Select Course and semester. | |
| | 3. Select Exam Type. | |
| | 4. Click “Analysis”. | |
| | | 5. Check the information and request for get Exam information’s. |

| | | |
|---|---|---|
| | | 6. Show “Exam information’s”. |
| **Critical Scenarios** | • Not the exam time. | |

User interface prototype:

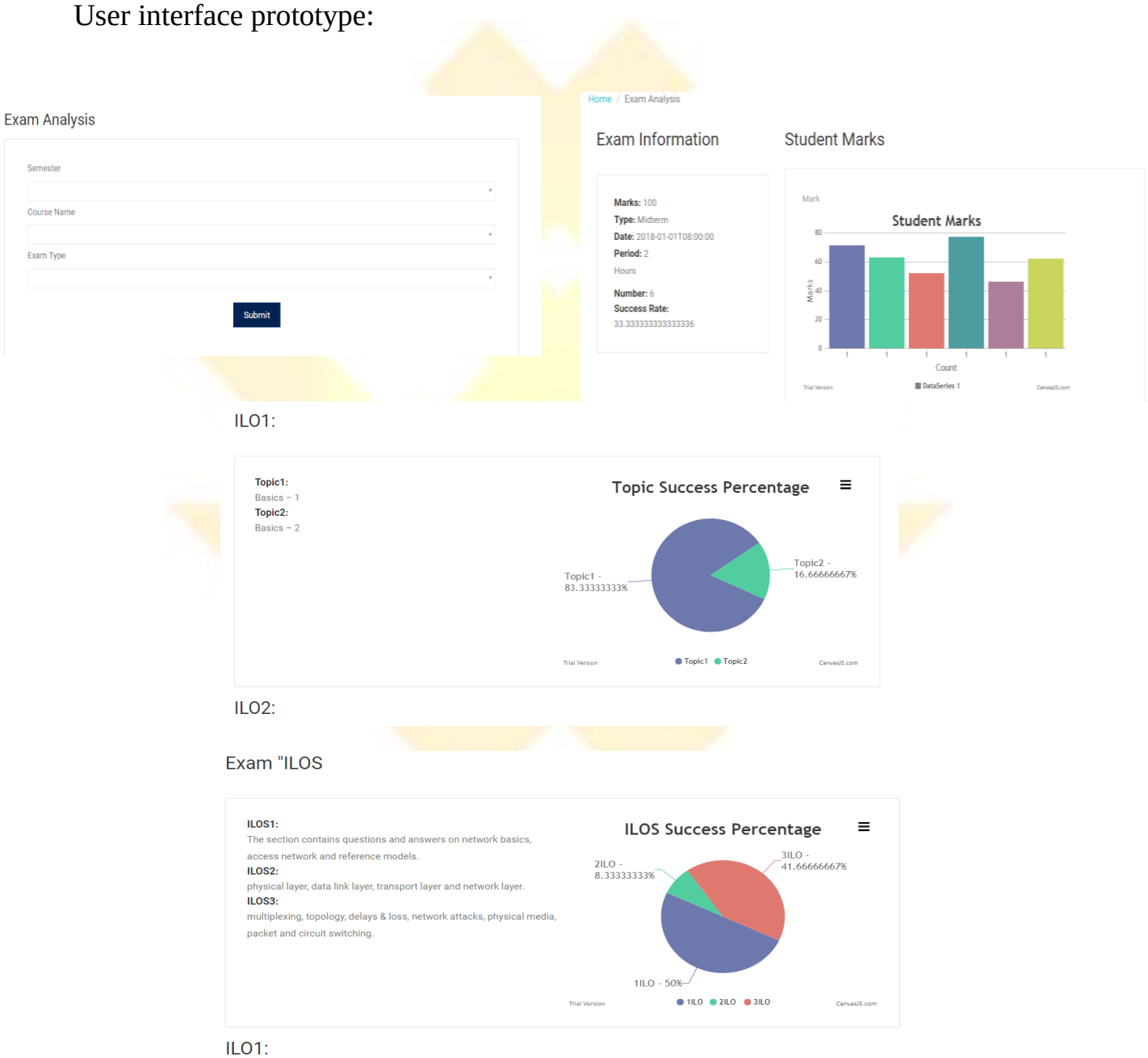



*Figure 21:Exam Analysis UI Prototypes*

| UC-B-12 | Course Analysis | |
|---|---|---|
| **Actor** | Admin. | |
| **Brief Description** | This use case describes how Admin can Analysis any Course. | |
| **Pre-Conditions** | • Logged in as Admin. | |
| **Post-conditions** | • Get information about Exam like Course like Marks, percentage. | |
| **Flow of events** | **Actor** | **System** |
| | 1. Click “Course Analysis”. | |
| | 2. Select semester. | |
| | 3. Select Course. | |
| | 4. Click “Analysis”. | |
| | | 5. Check the information and request for get Course information’s. |
| | | 6. show “Course information’s”. |
| **Critical Scenarios** | • No information about Course. | |

*Table 20:UC-B-12 Course Analysis*

# User Interface prototype:

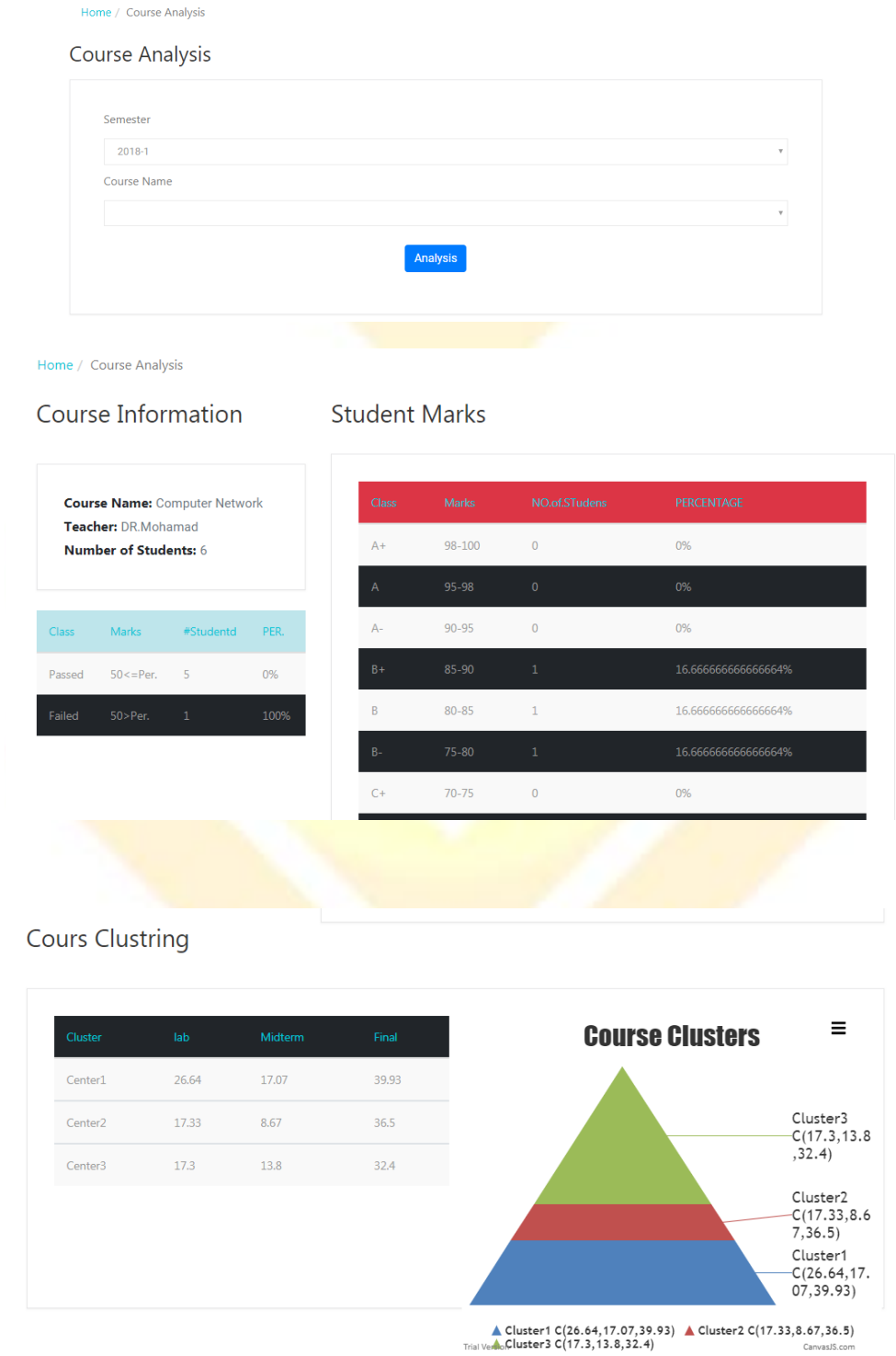



*Figure 22:Course Analysis UI Prototypes*

# C.OES API

*Table 21:UC-C-C1 Fetch the first question*

| UC-C-1 | Fetch the first question | |
|---|---|---|
| **Actor** | MVC Client. | |
| **Brief Description** | This use case describes how the system can request for Start new exam and get first question from the web API server. | |
| **Pre-Conditions** | • System request for start new exam and get question the first from web API. | |
| **Post-conditions** | • Web API save exam information in the database and Apple question algorithm. | |
| **Flow of events** | **Actor** | **System** |
| | 1. Request for new exam". | |
| | | 2. Select exam information. |
| | | 3. Save the exam information in the database. {student_id, subject_id}. |
| | | 4. Apply Question Algorithm to select first question. |
| | | 5. send the first question Refer to the start of the exam and register its information. |
| **Critical Scenarios** | • The Web API server stopped.<br>• No connection to Web API server. | |

*Table 22:UC-C-2 Correct and fetch the next question*

| UC-C-2 | Correct and fetch the next question | |
|---|---|---|
| **Actor** | MVC Client. | |
| **Brief Description** | This use case describes how the system can request for next question from the web API server. | |
| **Pre-Conditions** | • System request for the next question from web API. | |
| **Post-conditions** | • Web API server correct student answer, save it in the database and get the next question. | |
| **Flow of events** | **Actor** | **System** |
| | 1. Pass student answer with question id. | |
| | | 2. Connect to database and get the question true answer. |
| | | 3. Apply auto correction algorithm to Correct this answer only. |
| | | 4. Save the information in the database. |
| | 5. Request for next question. | |
| | | 6. Randomly selects from the Student Questions list. |
| | | 7. Appropriate question Depending on question algorithm. |
| **Critical Scenarios** | • The Web API server stopped.<br>• No connection to Web API server. | |

*Table 23:UC-C-3 Fetch the final Result*

| UC-C-3 | Fetch the final Result | |
|---|---|---|
| **Actor** | MVC Client. | |
| **Brief Description** | This use case describes how the system get student final result from the web API server and stop the exam. | |
| **Pre-Conditions** | • System request for the final result from web API. | |
| **Post-conditions** | • Web API server end the exam and calculate the final result and save it in the database. | |
| **Flow of events** | **Actor** | **System** |
| | 1. Request for final result". | |
| | | 2. Server connect to database to get student individually answers. |
| | | 3. Apply Auto correction algorithm to calculate the final result. |
| | | 4. Save the student final result in the Database. |
| | | 5. Send the final result with details to the system. |
| **Critical Scenarios** | • The Web API server stopped.<br>• No connection to Web API server. | |

*Table 24:UC-C-4 Get Available view*

| UC-C-4 | Get Available view |
|---|---|
| **Actor** | MVC Client. |

| | | |
|---|---|---|
| **Brief Description** | This use case describes how the system can request for the view that Customized for each user. | |
| **Pre-Conditions** | • System request for View from web API | |
| **Post-conditions** | • Web API return view Determined according to User login information. | |
| **Flow of events** | **Actor** | **System** |
| | 1. Request for View". | |
| | | 2. Check login information. |
| | | 3. According to User Role The API select Appropriate view. |
| | | 4. Return view. |
| **Critical Scenarios** | • The Web API server stopped.<br>• No connection to Web API server. | |

# D.EAS API

*Table 25: UC-D-1 Get Exam Information*

| UC-D-1 | Get Exam Information |
|---|---|
| **Actor** | MVC Client. |
| **Brief Description** | This use case describes how the system can request for Analysis the exam and get information about students marks and course ILOS, Topic. |
| **Pre-Conditions** | • System request for Analysis Exam from web API. |
| **Post-conditions** | • Web API Analysis the exam and return the information to MVC. |

| | | |
|---|---|---|
| **Flow of events** | **Actor** | **System** |
| | 1. Request for Analysis the Exam". | |
| | | 2. Get Exam ID. |
| | | 3. Start Analysis the exam and found the ILOS and topic. |
| | | 4. Save result in list. |
| | | 5. Send result to MVC. |
| **Critical Scenarios** | • The Web API server stopped.<br>• No connection to Web API server. | |

*Table 26: UC-D-2 Get Course Information*

| **UC-D-2** | **Get Course Information** | |
|---|---|---|
| **Actor** | MVC Client. | |
| **Brief Description** | This use case describes how the system can request for get course information. | |
| **Pre-Conditions** | • System request for Analysis Course from web API. | |
| **Post-conditions** | • Web API Analysis the Course and return the information to MVC. | |
| **Flow of events** | **Actor** | **System** |
| | 1. Request for Analysis the Course". | |
| | | 2. Get Course ID. |

| | | |
|---|---|---|
| | | 3. Start Analysis the Course and Apply k-means. |
| | | 4. Save result in list. |
| | | 5. Send result to MVC. |
| **Critical Scenarios** | • The Web API server stopped.<br>• No connection to Web API server. | |

*Table 27:UC-D-3 Teacher Course Information*

| **UC-D-3** | **Teacher Course Information** | |
|---|---|---|
| **Actor** | MVC Client. | |
| **Brief Description** | This use case describes how the system can request for Analysis specific teacher Course. | |
| **Pre-Conditions** | • System request for Analysis specific teacher Course. | |
| **Post-conditions** | • Web API Analysis the Course and return the information to MVC. | |
| **Flow of events** | **Actor** | **System** |
| | 1. Request for Analysis the Course". | |
| | | 2. Get Course ID. |
| | | 3. Start Analysis the Course and calculate Max mark, Min mark and Average mark. |
| | | 4. Save result in list. |

| | | |
|---|---|---|
| | | 5. Send result to MVC. |
| **Critical Scenarios** | • The Web API server stopped.<br>• No connection to Web API server. | |

## 1.2. Non-Functional Requirements

By using MVC as our development language we obtain lots of implicit nonfunctional requirements because MVC stands for Model, View and Controller and is a more than reliable alternative when it comes to creating and implementing systems in which user interfaces are included. Use of this technique provides advantages such as increase reliability and robustness in the software, which increases its life cycle; code reuse; separation of concepts and greater ease

### 1.2.1. Speed and Latency

Because of our project is a distributed system and the examination process is happening in the real time we spend a lot of time testing our algorithms with big data set to make sure that our engine works in the real time and in order to achieve this result we optimize and reduce their complexity by using fast data structures.

### 1.2.2. Security

As online exam system one of our project special points is security. our system must be secured so that lead us to new question. What are the risks that our system may face in the future??

There are too many cyber-attacks may cause the system failure or (system down) The following attacks are: We protected our system from three types of the attack.

### 1.2.2.1. Denial-of-service

a denial-of-service attack (DoS attack) is a cyber-attack where the perpetrator seeks to make a machine or network resource unavailable to its intended users by temporarily or indefinitely disrupting services of a host connected to the Internet. Denial of service is typically accomplished by flooding the targeted machine or resource with superfluous requests in an attempt to overload systems and prevent some or all legitimate requests from being fulfilled. A DoS attack is analogous to a group of people crowding the entry door or gate to a shop or business, and not letting legitimate parties enter into the shop or business, disrupting normal operations.

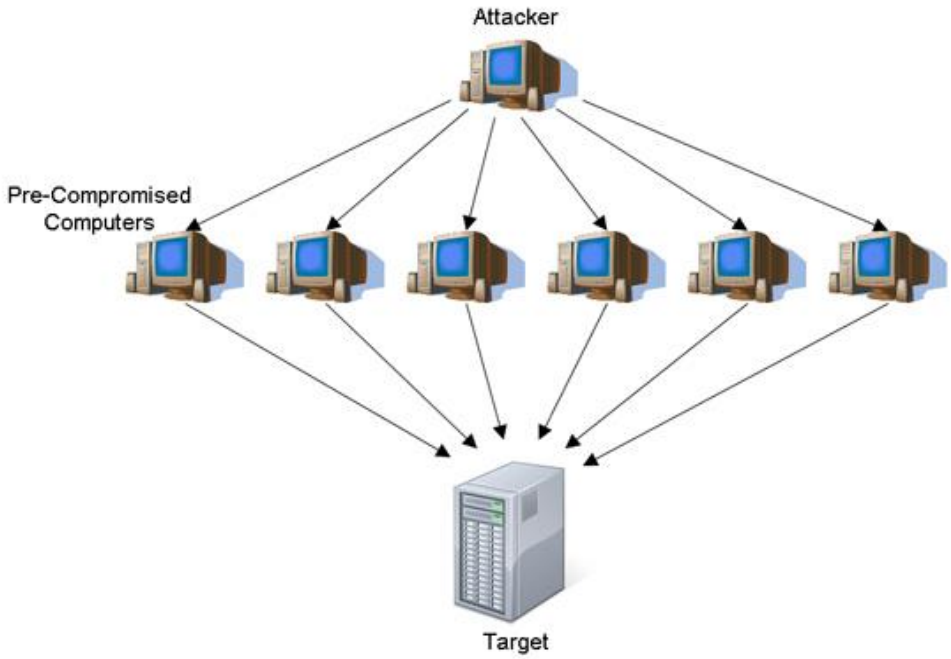



*Figure 23:Denial of Service attack*

**Counter measures against DoS:**

The only thing you can do to prevent such an attack is to block the response to the attackers. You have no control over the requests, so you have to catch the attacker as early as possible after the request has been received by the web server.

There are two challenges to blocking the attacks

1. Identify the attackers
2. Block the response only to the attackers

To catch the request as early as possible, an HttpModule is the right place. It is executed before any page or any other handler so the impact on the server can be minimized. This HttpModule monitors all requests and block requests coming from IP addresses that make many requests in a short period of time. After a while the attacking IP address gets released from blocking.

The module is a high performance and lightweight protection from DoS attacks and very easy to implement.

### 1.2.2.2. Data Encryption

- **Data Encryption Standard -DES**

## Introduction:

As mentioned earlier there are two main types of cryptography in use today **symmetric** or **secret key** cryptography and **asymmetric** or **public key** cryptography. Symmetric key cryptography is the oldest type whereas asymmetric cryptography is only being used publicly since the late 1970's1. Asymmetric cryptography was a major milestone in the search for a perfect encryption scheme. Secret key cryptography goes back to at least Egyptian times and is of concern here. It involves the use of only one key which is used for both

encryption and decryption (hence the use of the term symmetric). Depicts this idea. It is necessary for security purposes that the secret key never be revealed. Several types of symmetric algorithms are used today. They have different methods of providing encryption and decryption functionality.

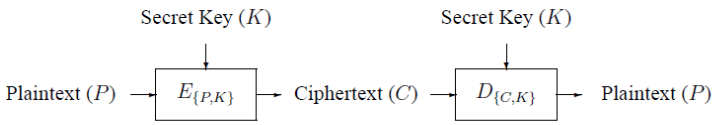



*Figure 24:DES Architecture*

## How does DES work?

- DES is a symmetric block encryption algorithm. When 64-bit blocks of plaintext go in, 64-bit blocks of cipher text come out.
- It is also a symmetric algorithm, meaning the same key is used for encryption and decryption.
- It uses a 64-bit key: 56 bits make up the true key, and 8 bits are used for parity.
- When the DES algorithm is applied to data, it divides the message into blocks and operates on them one at a time.
- The blocks are put through 16 rounds of transposition and substitution functions.
- The result is 64-bit blocks of ciphertext.

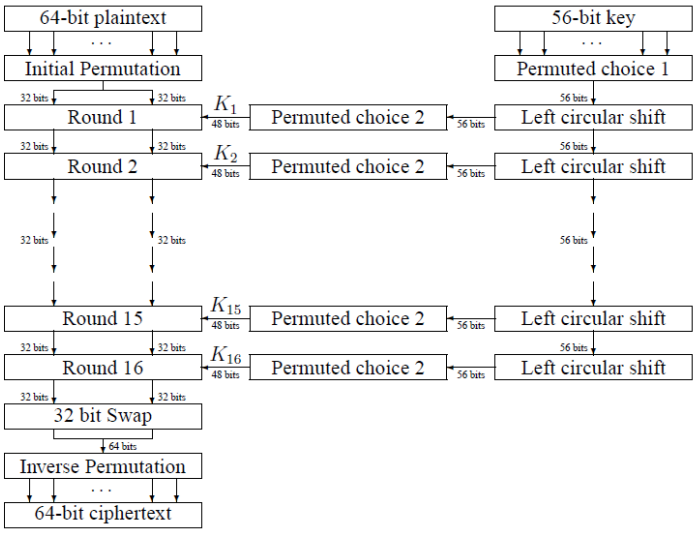



*Figure 25:DES Detailed Steps*

## DES Modes:

Block ciphers have several modes of operation. Each mode specifies how a block cipher will operate.

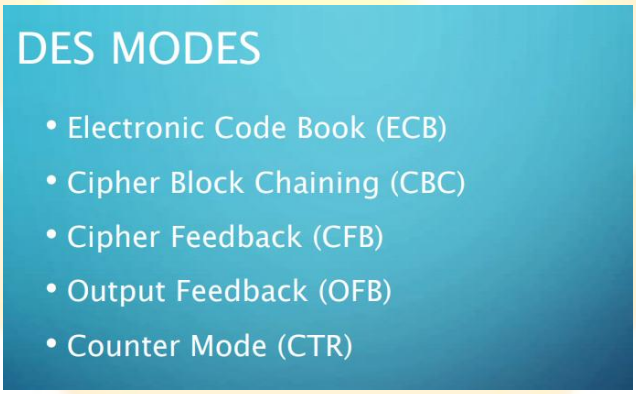



*Figure 26:DES Modes of Operation*

## How we apply DES in our system?

When the teacher wants to add a new question, the system applies Des. Encryption on the question text and question answers before save it in the Database, these steps is too important to protect Questions from hacking

To apply DES Algorithm we use ASP DESCryptoServiceProvider, so the teacher inserts question and his answer, the system call Des. Encryption For encryption now in the Database the question was saved in encrypted form:

AL-DOON-PC\SQLE...o.Questions_Bank

| Question_ID | Topic_ID | Question | Expected_Time | Difficulty_Level | Question_Freq... | Thinking_Skills | Question_Mark |
|---|---|---|---|---|---|---|---|
| 3 | 1 | 5+Isie3b/GrHo... | 3 | 1 | 0 | Writing | 1 |
| 4 | 1 | rDLEX84Qwbj3I... | 4 | 2 | 0 | Writing | 2 |
| 5 | 1 | BtxVCzsIcQ3fH... | 5 | 3 | 0 | Reading | 3 |
| 6 | 1 | V5qqNzdY6PxA... | 6 | 4 | 0 | Critical Thiking | 4 |
| 7 | 1 | Z+5QhZiP+uH... | 7 | 5 | 0 | Designing | 5 |

| Answer_ID | Question_ID | Answer | is_True |
|---|---|---|---|
| 6 | 3 | y9QfJvVT3M4= | True |
| 7 | 3 | MK579F7nrd0= | False |
| 8 | 3 | GgCkjOTHM3s= | False |
| 9 | 3 | TpJcg1VBMHec... | False |
| 10 | 4 | FF5oK9HZaz8= | True |
| 11 | 4 | 9gSQ3H1bBT8= | False |

*Figure 27:Encrypted Sample of the DB*

‘

## When we use DES Decrypt?

In the Exam. after Question algorithm Controller prepare exam question, the system applies DES. Decrypt and display questions to students, so from save questions in database to Exam date, questions Safely reserved

## RSA (Rivest–Shamir–Adleman):

Is one of the first public-key cryptosystems and is widely used for secure data transmission. In such a cryptosystem, the encryption key is public and it is different from the decryption key which is kept secret (private). In RSA, this asymmetry is based on the practical difficulty of the factorization of the product of two large prime numbers, the "factoring problem". The acronym RSA is made of the initial letters of the surnames of Ron Rivest, Adi Shamir, and Leonard Adleman, who first publicly described the algorithm in 1978. Clifford Cocks, an English mathematician working for the British intelligence agency Government Communications Headquarters (GCHQ), had developed an equivalent system in 1973, but this was not declassified until 1997.

A user of RSA creates and then publishes a public key based on two large prime numbers, along with an auxiliary value. The prime numbers must be kept secret. Anyone can use the public key to encrypt a message, but with currently published methods, and if the public key is large enough, only someone with knowledge of the prime numbers can decode the message feasibly. Breaking RSA encryption is known as the RSA problem. Whether it is as difficult as the factoring problem remains an open question.

RSA is a relatively slow algorithm, and because of this, it is less commonly used to directly encrypt user data. More often, RSA passes encrypted shared keys for symmetric key cryptography which in turn can perform bulk encryption-decryption operations at much higher speed.

## Operation

The RSA algorithm involves four steps: key generation, key distribution, encryption and decryption.

A basic principle behind RSA is the observation that it is practical to find three very large positive integers e, d and n such that with modular exponentiation for all integer m (with $0 \leq m < n$):

$(m^e)^d = m \pmod n$

and that even knowing e and n or even m it can be extremely difficult to find d.

In addition, for some operations it is convenient that the order of the two exponentiations can be changed and that this relation also implies:

$(m^d)^e = m \pmod n$

RSA involves a public key and a private key. The public key can be known by everyone, and it is used for encrypting messages. The intention is that messages encrypted with the public key can only be decrypted in a reasonable amount of time by using the private key. The public key is represented by the integers n and e; and, the private key, by the integer d (although n is also used during the decryption process. Thus, it might be considered to be a part of the private key, too). m represents the message (previously prepared with a certain technique explained below).

### Key generation:

The keys for the RSA algorithm are generated the following way:

- Choose two distinct prime numbers p and q.
  For security purposes, the integers p and q should be chosen at random and should be similar in magnitude but differ in length by a few digits to make factoring harder. Prime integers can be efficiently found using a primality test.
- Compute n = p. q.
  n is used as the modulus for both the public and private keys. Its length, usually expressed in bits, is the key length.

- Compute $\lambda(n) = \text{lcm}(\lambda(p), \lambda(q)) = \text{lcm}(p - 1, q - 1)$, where $\lambda$ is Carmichael's totient function. This value is kept private.
- Choose an integer e such that $1 < e < \lambda(n)$ and GCD $(e, \lambda(n)) = 1$; i.e., e and $\lambda(n)$ are coprime.
  Determine d as $d \equiv e^{-1} \pmod{\lambda(n)}$; i.e., d is the modular multiplicative inverse of e (modulo $\lambda(n)$).

**How R SA is applied in our system:**

Since RSA is relatively slow, we used RSA encryption to encrypt the DES secret key and pass it. The following figure explains How our encryption works.

# 2. Software Analysis

## 2.1. Collaboration Diagram

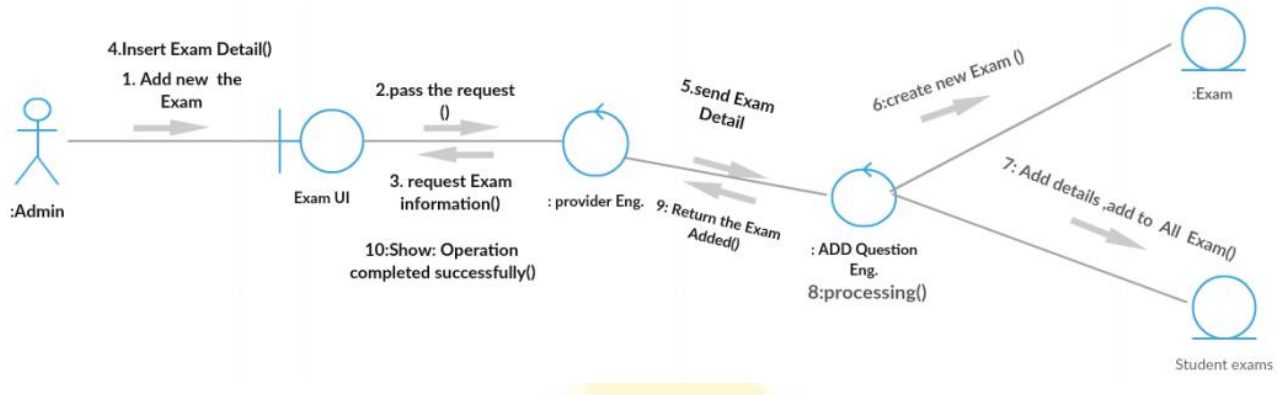



*Figure 28:Collaboration-Create Exam*

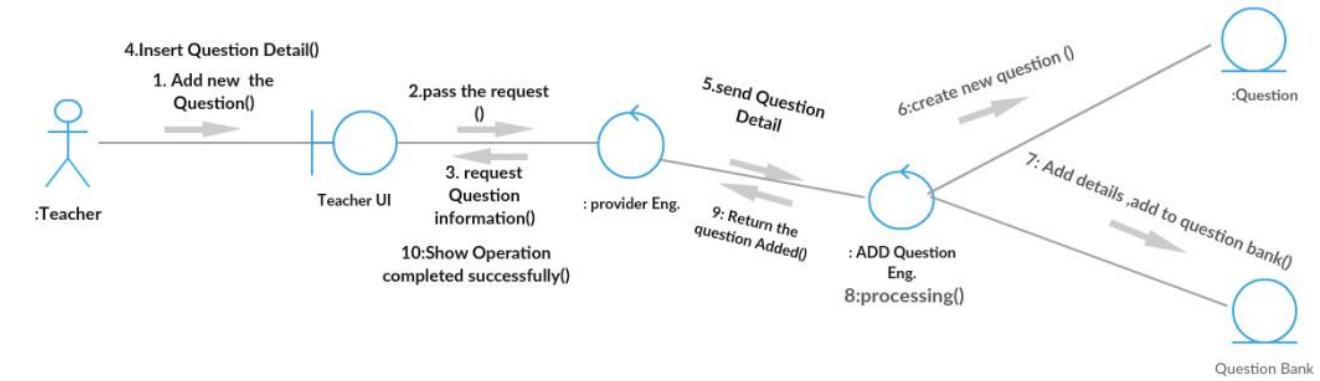



*Figure 29:Collaboration-Add New Question*

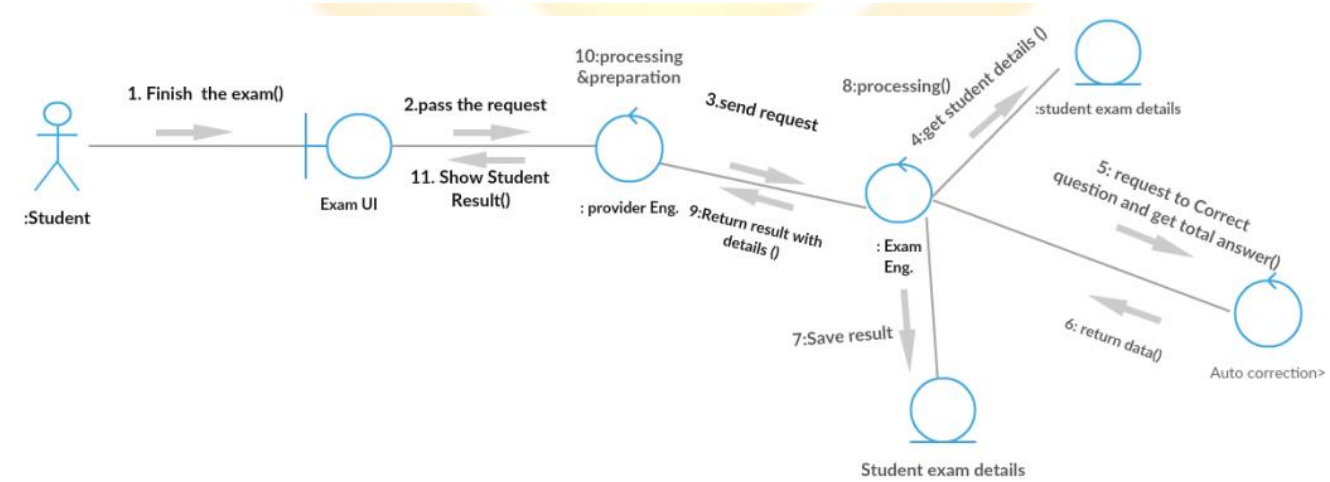



*Figure 30:Collaboration-Finish Exam*

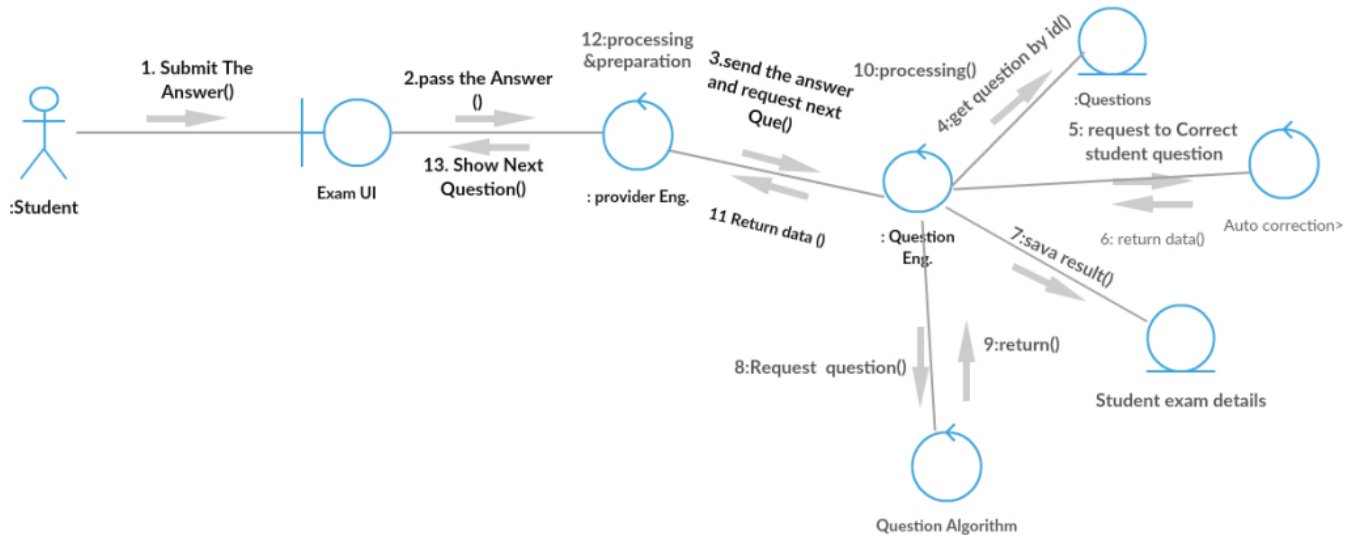



*Figure 31:Collaboration-Question Correction*

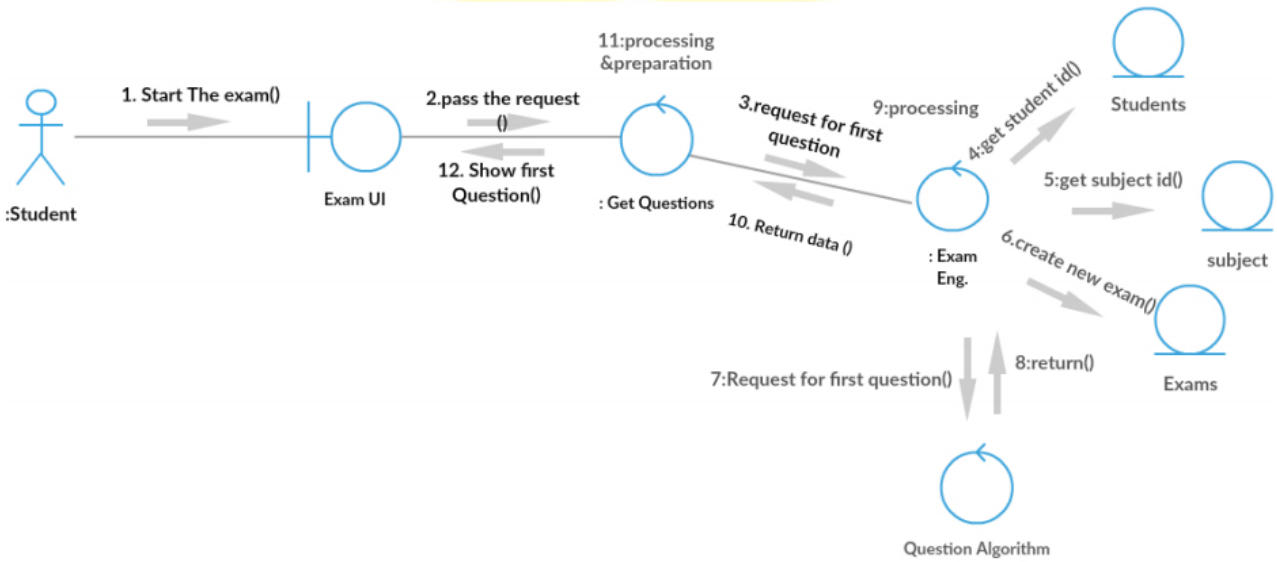



*Figure 32:Collaboration-Start Exam*

## 2.2. CRC Cards

*Table 28:CRC Main*

| Boundary | Controller | Entity |
|---|---|---|
| Account Viewer | MVC-Account Controller | User |
| Email Register Viewer | MVC-Student Controller | Questions Bank |
| Homepage Viewer | MVC-Admin Controller | Question Answers |
| Start Exam Button | MVC-Teacher Controller | Course |
| New Question Button | API-Teacher Controller | Students Marks |
| Previous Question Button | API-Admin Controller | Exam |
| Submit Question Button | API-Student Controller | Topic |
| Finish Exam Button | API-Account Controller | ILOS |
| My Mark Button | | Semester Course |
| GPA Button | | Students Answer |
| Add Question Button | | Calander |
| Generate Exam Button | | Students Marks |
| Add User | | |
| Create Exam | | |
| Exam Analysis | | |
| Course Analysis | | |
| Student Analysis | | |
| Semester Analysis | | |

## ❖ Boundary

*Table 29:CRC Boundary- Account Viewer*

| CRC Card | |
|---|---|
| **Class Name: Account Viewer** | |
| **Responsibility** | **Collaborations** |
| **Signup request** | MVC-Account Controller |
| **Login request** | MVC-Account Controller |
| **Logout request** | MVC-Account Controller |

*Table 30:CRC Boundary-Email Register*

| CRC Card | |
|---|---|
| **Class Name: Email Register** | |
| **Responsibility** | **Collaborations** |
| **Open register window** | **-** |
| **Fill user information** | MVC-Account Controller |
| | |

*Table 31:CRC Boundary- Homepage Viewer*

| CRC Card | |
|---|---|
| **Class Name: Homepage Viewer** | |
| **Responsibility** | **Collaborations** |
| **Open homepage** | **-** |
| | |

*Table 32:CRC Boundary- Exam page Viewer*

| CRC Card | |
|---|---|
| **Class Name: Exam page Viewer** | |
| **Responsibility** | **Collaborations** |
| **Open Exam page** | MVC-Exam Controller |
| **Question request** | API-Question Controller |

*Table 33:CRC Boundary- Navigating question Button*

| CRC Card | |
|---|---|
| **Class Name: Navigating question Button** | |
| **Responsibility** | **Collaborations** |
| **Next Question request** | API-Question Controller |
| **Previous Question request** | API-Question Controller |

*Table 34:CRC Boundary- Finish Exam Button*

| CRC Card | |
|---|---|
| **Class Name: Finish Exam Button** | |
| **Responsibility** | **Collaborations** |
| **correct student answers request** | API-Question Correction Controller |
| **Student result request** | API-Question Correction Controller |

*Table 35:CRC Boundary- Exam Result page*

| CRC Card | |
|---|---|
| **Class Name: Exam Result page** | |
| **Responsibility** | **Collaborations** |
| **Open Exam Result Page** | - |
| **Fill student answers** | API-Question Correction Controller |

*Table 36:My marks button*

| CRC Card | |
|---|---|
| **Class Name: My marks button** | |
| **Responsibility** | **Collaborations** |
| **request Marks Page** | MVC-Student Controller |
| **Fill student marks** | API-Student Controller |

*Table 37:CRC Boundary-GPA button*

| CRC Card | |
|---|---|
| **Class Name: GPA button** | |
| **Responsibility** | **Collaborations** |
| **request GPA Page** | MVC-Student Controller |
| **Fill student marks** | API-Student Controller |

*Table 38: CRC Boundary-Add question view*

| CRC Card | |
|---|---|
| **Class Name: Add question view** | |
| **Responsibility** | **Collaborations** |
| **open Add question Page** | MVC-Teacher Controller |
| **Fill question information** | MVC-Teacher Controller |

*Table 39: CRC Boundary-Add question button*

| CRC Card | |
|---|---|
| **Class Name: Add question button** | |
| **Responsibility** | **Collaborations** |
| **Add question request** | API-Teacher Controller |

*Table 40: CRC Boundary-Create Exam view*

| CRC Card | |
|---|---|
| **Class Name: Create Exam view** | |
| **Responsibility** | **Collaborations** |
| **open Create Exam Page** | MVC-Teacher Controller |
| **Fill Exam pattern** | MVC-Teacher Controller |

*Table 41: CRC Boundary-Generate Exam button*

| CRC Card | |
|---|---|
| **Class Name: Generate Exam button** | |
| **Responsibility** | **Collaborations** |
| **Create Exam request** | API-Teacher Controller |

*Table 42: CRC Boundary-Exam Analysis view*

| CRC Card | |
|---|---|
| **Class Name: Exam Analysis view** | |
| **Responsibility** | **Collaborations** |
| **Open Exam Analysis view** | API-Teacher Controller |

*Table 43: CRC Boundary-Exam Analysis button*

| CRC Card | |
|---|---|
| **Class Name: Exam Analysis button** | |
| **Responsibility** | **Collaborations** |
| **Exam Analysis request** | API-Admin Controller |

## ❖ Controller

*Table 44: CRC Controller-API-Account Controller*

| CRC Card | |
|---|---|
| **Class Name: API-Account Controller** | |
| **Responsibility** | **Collaborations** |
| **Login request** | User |
| **Validate account** | - |
| **Open session** | |
| **Logout request** | |
| **Signup request** | User |
| **Validate data** | User |

*Table 45: CRC Controller- API-Question Controller*

| CRC Card | |
|---|---|
| **Class Name: API-Question Controller** | |
| **Responsibility** | **Collaborations** |
| **Request questions** | Question Bank |
| **Request Next question** | Question Bank |
| **Request Previous question** | Question Bank |

*Table 46: CRC Controller- API-Question Correction Controller*

| CRC Card | |
|---|---|
| **Class Name: API-Question Correction Controller** | |
| **Responsibility** | **Collaborations** |
| **Request answer correct** | Question Answers |
| **Request student result** | Question Answers |

*Table 47: CRC Controller- API-Student Controller*

| CRC Card | |
|---|---|
| **Class Name: API-Student Controller** | |
| **Responsibility** | **Collaborations** |
| **Request Student mark** | Student course marks |
| **Request student GPA** | Student Semester |

*Table 48: CRC Controller- API-Teacher Controller*

| CRC Card | |
|---|---|
| **Class Name: API-Teacher Controller** | |
| **Responsibility** | **Collaborations** |
| **Request add question mark** | Questions Bank |
| **Request create exam** | Exam |

*Table 49: CRC Controller- API-Admin Controller*

| CRC Card | |
|---|---|
| **Class Name: API-Admin Controller** | |
| **Responsibility** | **Collaborations** |
| **Request add user mark** | User |
| **Request create exam** | Exam |

*Table 50: CRC Controller- API-Analysis Admin Controller*

| CRC Card | |
|---|---|
| **Class Name: API-Analysis Admin Controller** | |
| **Responsibility** | **Collaborations** |
| **Request Exam analysis** | Exam |
| **Request course analysis** | Course |

## ❖ Entity

*Table 51: CRC Entity-User*

| CRC Card | |
|---|---|
| **Class Name: User** | |
| **Responsibility** | **Collaborations** |
| **Get email & password** | - |
| **Check email & password** | - |
| **Save user data** | - |
| **Check user email** | - |
| **Get user information** | - |

*Table 52: CRC Entity-User Attribute*

| Attribute |
|---|
| **FirstName, Last Name, Email, Address, Birthday** |

*Table 53: CRC Entity-Questions*

| CRC Card | |
|---|---|
| **Class Name: Questions** | |
| **Responsibility** | **Collaborations** |
| **Get question** | **-** |
| **Add question** | **-** |
| **Edit question** | **-** |
| **Delete question** | **-** |

*Table 54: CRC Entity-Questions Attribute*

| Attribute |
|---|
| **Question, Topic, Thinking skills, difficulty, mark** |

*Table 55: CRC Entity-Course*

| CRC Card | |
|---|---|
| **Class Name: Course** | |
| **Responsibility** | **Collaborations** |
| **Get course** | **-** |
| **Add course** | **-** |
| **Edit course** | **-** |

*Table 56: CRC Entity-Course Attribute*

| Attribute |
|---|
| **Name, Credit hours, description** |

*Table 57: CRC Entity-ILOS*

| CRC Card | |
|---|---|
| **Class Name: ILOS** | |
| **Responsibility** | **Collaborations** |
| **Get ILO** | **-** |
| **Add ILO** | **-** |

*Table 58: CRC Entity-ILOS Attribute*

| Attribute |
|---|
| **ILO, description** |

# 3. Software Design

## 3.1. Database Design (ERD)

### 3.1.1. Identity ERD

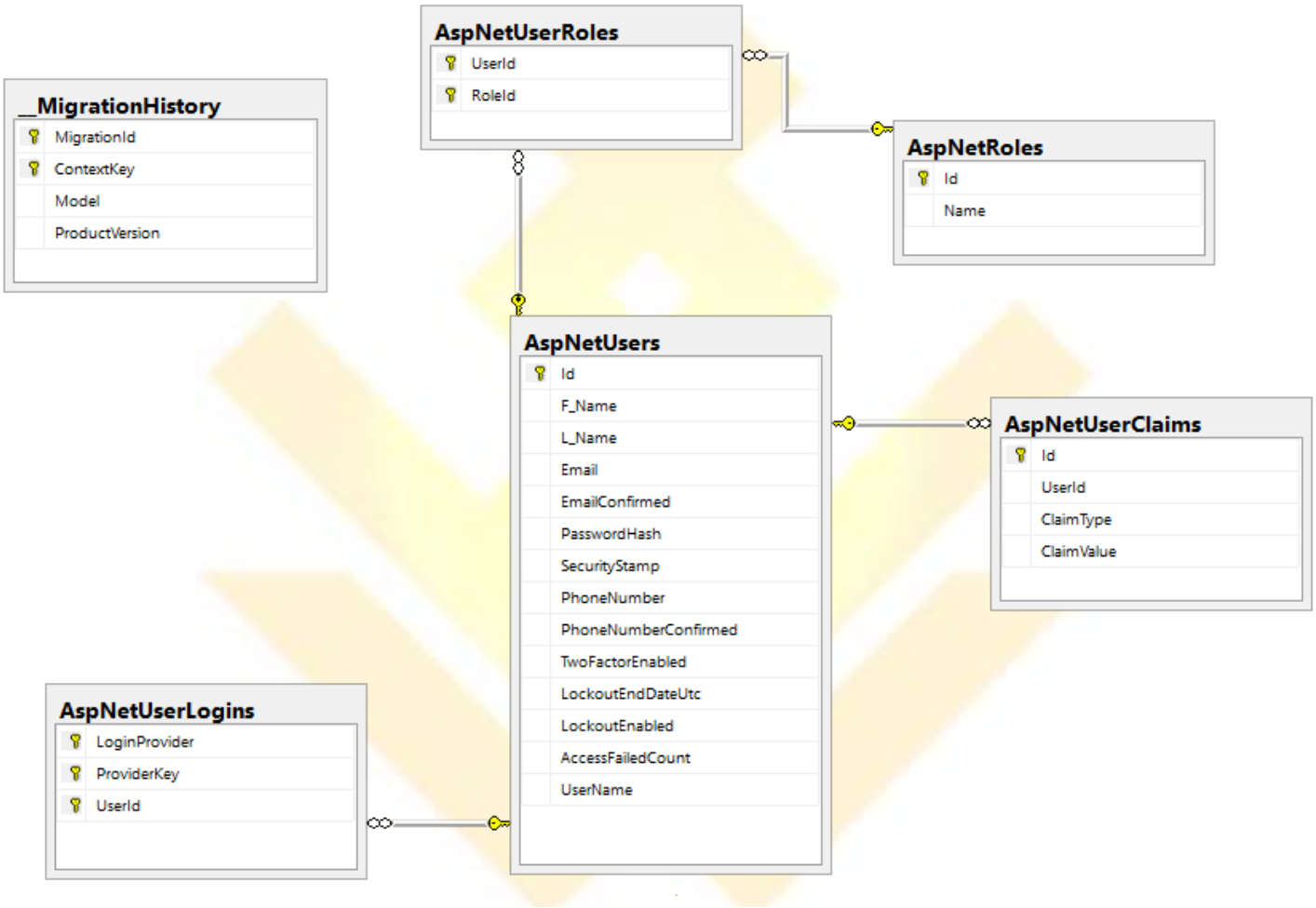



*Figure 33:Identity ERD*

## 3.1.2. EAS Database

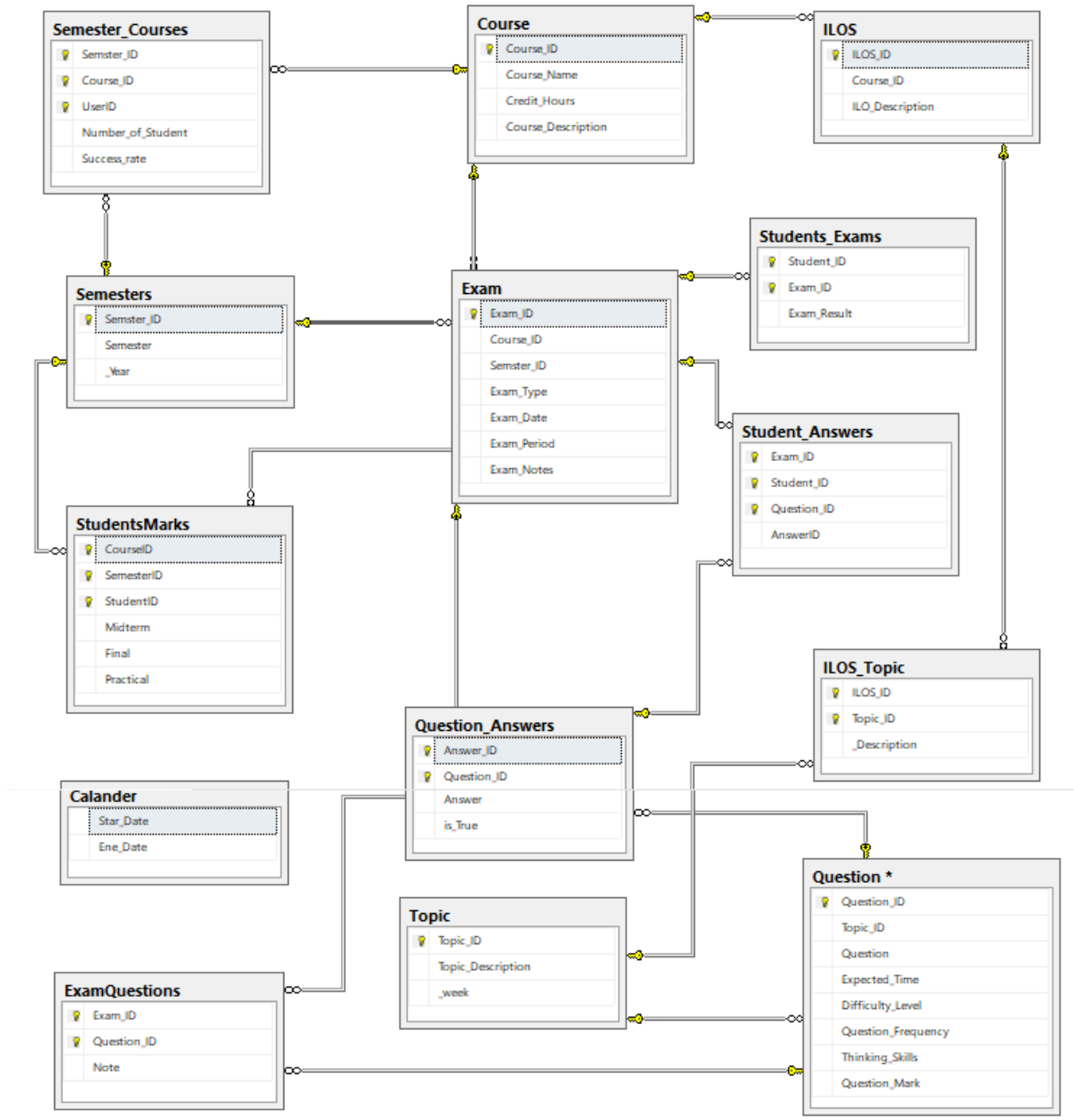



*Figure 34:EAS ERD*

# 4. Testing

## 4.1. Test suits

*Table 59: Test Suit - Fetch Next Question*

| A test suit to verify the "Fetch Next Question" feature | | | | | | |
|---|---|---|---|---|---|---|
| | | TSI | | | | |
| **Title** | | Verify "Fetch Next Question" functionality | | | | |
| **Description** | | To test the different scenarios that might arise while a user is trying to login | | | | |
| | | | | | | |
| # | Summary | Dependency | Pre-condition | Post-Condition | Execution Steps | Executed Output |
| **TC1** | Verify that user already logged in to the system can Fetch Question | | Student who logged in type a message and click send | The Question is Fetched and Viewed | 1. Login as student<br>2. Click next Question | New Question appeared to the student |
| **TC2** | Verify That Student Can't Fetch Question after the end of time | | Student who logged in Click Fetch Next Question after the end of exam time | The Question is not Fetched and Error message appeared | 1. Login as student<br>2. Wait until the time of Exam is finished<br>3. Click Fetch Question | Pop out Error Message tell the user that the exam ended and can't get another question |
| **TC3** | Verify that question Difficulties is reasonable | | Student Click Next Question | The Fetched Question difficulty is reasonable | 1. Login as student<br>2. Click next question | New question appeared to the student |

| A test suit to verify the "Finish Exam Functionality" feature | | | | | | |
|---|---|---|---|---|---|---|
| | TSI | | | | | |
| **Title** | Verify "Finish Exam Functionality" functionality | | | | | |
| **Description** | To test the different scenarios that might arise while a user is trying to login | | | | | |
| | | | | | | |
| # | Summary | Dependency | Pre-condition | Post-Condition | Execution Steps | Executed Output |
| **TC1** | Verify that exam is finished after clicking finish exam button | | Student who logged in click finish exam | The exam is finished | 1. Login as student<br>2. Click finish exam | Pop out screen tell the Student that the exam is finished and His result |
| **TC4** | Verify the exam is Automatically finished after the time is ended | | The time of the exam is ended | The exam is finished | 1. Login as student<br>2. Wait until the time of Exam is finished | Pop out screen tell the Student that the exam is finished and His result |
| **TC3** | Verify that the result is showed after the exam is finished | TC1 or TC2 | The time of the exam is ended **OR** Student who logged in click finish exam | The exam is finished and the result is showed | 1. Login as student<br>2. Wait until the time of Exam is finished **OR** the Finish exam button is clicked | Pop out screen tell the Student that the exam is finished and His result |

*Table 60: Test Suit - Finish Exam Functionality*

# 4th Chapter

**(Requirements & Design)**

# 1. Algorithms

## 1.1. Grandy's Number

The purpose to build online exam system is to find an algorithm doing the following tasks:

1. Degree of difficulty of the Questions pattern must be fair to all students.
2. The Questions pattern should be Different from one student to another. To ensure that we designed question algorithm using the Grundy's Numbers.

### Grundy's Numbers

In 1939, Patrick Michael Grundy wrote a paper in which he defined the following game. Like in the game of NM, the game consists of stones (or matches, marbles, etcetera) distributed over several piles. A turn for one of the players consists of dividing one of these piles into two different-sized piles. For example, a pile consiting of 7 stones can be divided into a pile of 6 and a pile of 1, or a pile of 5 and a pile of 2 or …. However, it cannot be divided into two piles of 3. Whoever cannot divide any more piles (because all piles contain 1 or 2 stones) loses the game.

**Example:** The graph-representation of Grundy's Game starting with a pile of 7 stones are as follows.

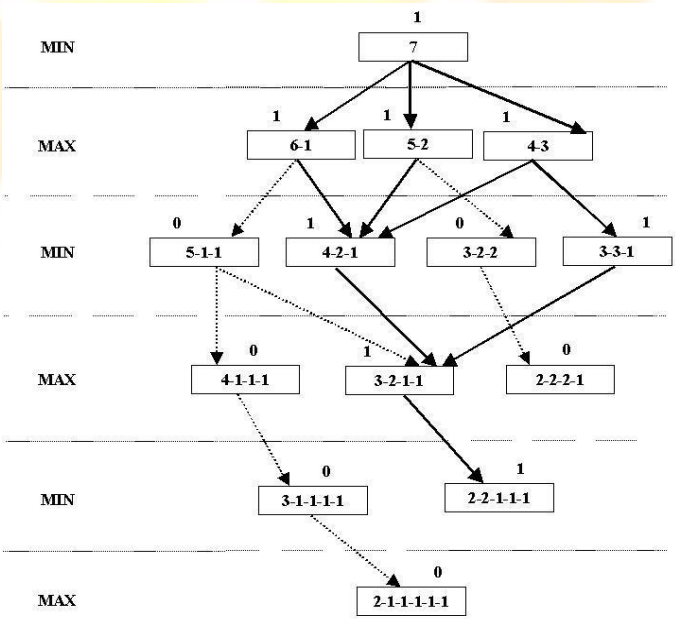



*Figure 35:Grunday's tree for 7*

**Steps of Questions Algorithm**

- when exam start, for each student the system. Randomly select degrees of difficulty from previous degrees.
- Find the next Grandy's Numbers in the next step and chose the path that have the maximum value.
- For each number the selected stack finds Grandy's numbers.
- Repeat Until the largest number in the stack is 2.

**Example:**

When student start the exam, the system selects randomly number 5 from this {5,4,3,2,1} and in next select the system selects from {4,3,2,1}, and next select

and each time the system selects a number it uses all its tree.

from

{3,2,1} ……

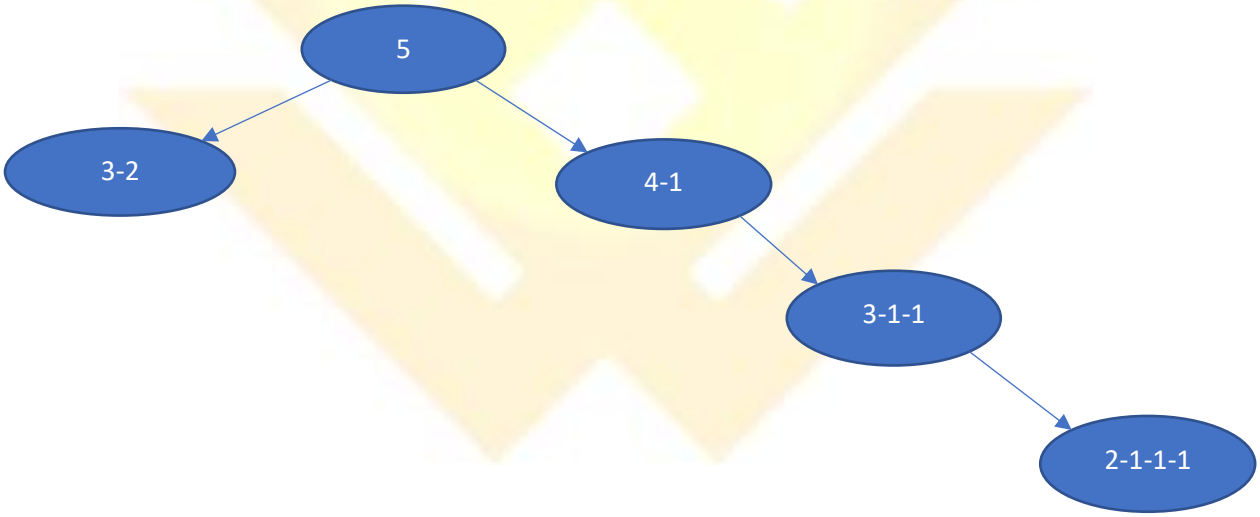



*Figure 36:Question Algorithm tree for 5*

According to the previous algorithm there are 5 question Will be submitted to the student for an answer:

1) questions with Difficulty >>>5

2) questions with Difficulty >>>4

3) questions with Difficulty>>> 1

4) questions with Difficulty >>>3

5) questions with Difficulty>>> 1

6) questions with Difficulty>>> 1

7) questions with Difficulty>>> 2

Etc.…

For each question, there is a number representing the repetition of choosing this question Each time this question is picked this number increases by one.

The system chooses the least frequent question to ensure that the questions are not repeated in the exam.

### 1.2. CSP Algorithm

## Constraint satisfaction problems (CSPs):

CSPs Are mathematical questions defined as a set of objects whose state must satisfy a number of constraints or limitations. CSPs represent the entities in a problem as a homogeneous collection of finite constraints over variables, which is solved by constraint satisfaction methods. CSPs are the subject of intense research in both artificial intelligence and operations research, since the regularity in their formulation provides a common basis to analyze and solve problems of many seemingly unrelated families. CSPs often exhibit high complexity, requiring a combination of heuristics and combinatorial search methods to be solved in a reasonable time. The Boolean satisfiability problem (SAT), the satisfiability modulo theories (SMT) and answer set programming (ASP) can be roughly thought of as certain forms of the constraint satisfaction problem.

The existence of a solution to a CSP can be viewed as a decision problem. This can be decided by finding a solution, or failing to find a solution after exhaustive search (stochastic algorithms typically never reach an exhaustive conclusion, while directed searches often do, on sufficiently small problems). In some cases the CSP might be known to have solutions beforehand, through some other mathematical inference process.

a constraint satisfaction problem is defined as a triple (X,D,C), where:

$$X = \{X_1, X_2, \ldots., X_n\}$$

$$D = \{D_1, D_2, \ldots., D_n\}$$

$$C = \{C_1, C_2, \ldots., C_n\}$$

where:

X are variables , D are Domains and C are constraints

How CSP is applied in our system:

**Variables:**

$V_1$: Topics, $V_2$: Difficulties, $V_3$: Thinking Skills, $V_4$: Time

**Domain**:

Dom($V_1$): {$Topic_1$, $Topic_2$, $Topic_3$ … $Topic_n$}

Dom($V_2$): {Very Easy, Easy, Average, Difficult, Very Difficult}

Dom($V_3$): {Recall, Analysis, Understanding, Compiling, Critical Thinking}

Dom($V_4$): {1min, 2min, 3min …etc.}.

**Constraints:**

- Total Number of Question =N where N is entered by the user **OR**
  Total Number of Question = N where N is satisfying.

$$\sum_{1}^{n} Q(V4) = TM$$

- $\sum_{1}^{n} Q(V_2) = 1$ & $\sum_{1}^{n} Q(V_4) = 1$sni

- $\sum_{1}^{n} Qt(V_2) = R_T$& $\sum_{1}^{n} Q(V_4) = R_{Ts}$

  Where $Q_t$ is questions for the current topic and $R_t$ is the ratio of the topic form the total wanted exam pattern.

## 1.3. K-means Clustering

**Introduction:**

Today the important challenge that higher education faces, is reaching a stage to facilitate the universities in having more efficient, effective and accurate educational processes. Data mining is considered as the most suited technology appropriate in giving additional insight into the lecturer, student,teacher, manager, and other educational staff behavior and acting as an active automated assistant in

helping them for making better decisions on their educational activities. As discussed before, lack of deep and enough knowledge in higher educational

system may prevent system management to achieve their quality objectives. Data mining technology can help bridging this knowledge gaps in higher educational system. Therefore, the hidden patterns, association and anomalies, which are discovered by some data mining techniques, can be used to improve the effectiveness, efficiency and the speed of the processes. As a result, this improvement may bring a lot of advantages to the higher educational system such as maximizing educational system efficiency, decreasing student's drop-out rate, increasing student's promotion rate, increasing student's retention rate, increasing student's transition rate, increasing educational improvement ratio, increasing student's success, increasing student's learning outcome, and reducing the cost of system processes. In order to achieve the above quality improvement, we need a data mining system that can provide the needed knowledge and insights for the decision makers in the higher educational system. We have analyzed such a system and in the next section we will present and describe an analysis model for this system.

**K-means:**

*K*-means clustering is a type of unsupervised learning, which is used when you have unlabeled data (i.e., data without defined categories or groups). The goal of this algorithm is to find groups in the data, with the number of groups represented by the variable *K*. The algorithm works iteratively to assign each data point to one of *K* groups based on the features that are provided. Data points are clustered based on feature similarity. The results of the *K*-means clustering algorithm a

1. The centroids of the *K* clusters, which can be used to label new data

2. Labels for the training data (each data point is assigned to a single cluster)

Rather than defining groups before looking at the data, clustering allows you to find and analyze the groups that have formed organically.

Each centroid of a cluster is a collection of feature values which define the resulting groups. Examining the centroid feature weights can be used to qualitatively interpret what kind of group each cluster represents.

The idea is to choose random cluster centers, one for each cluster. These centers are preferred to be as far as possible from each other. In this algorithm Euclidean distance measure is used between two multidimensional data points

$$X = (x1,x2,x3,\ldots\ldots\ldots xm )$$

$$Y = (y1,y2,y3,\ldots\ldots\ldots ym )$$

The Euclidean distance measure between the above points x and y are described as follows:

$$D(X, Y) = ( ! ( xi - yi) 2)1/2$$

**Algorithmic steps:**

**Input**: D = {d1, d2, d3, ..., dn } // Set of n data points. K - Number of desired clusters

**Output**: A set of K clusters.

**Method:**

1. Select the number of clusters. Let this number be k.
2. Pick k seeds as centroids of the k clusters. The seeds may be picked

randomly unless the user has some insight into the data.

3. Compute the Euclidean distance of each object in the dataset from each of the centroids.
4. Allocate each object to the cluster nearest, based on the distances computed in the previous step.
5. Compute the centroids of the clusters by computing the means of the attribute values if the objects are in each cluster.
6. Check if the stopping criterion has been met (e.g. the cluster membership is unchanged). If not go to step 3.

**Implementation of mining model:**

In this steps k-means clustering algorithm was applied to the proposed data an get valuable information, k- means is an old and most widely used by clustering algorithm.

1. Select K points as the initial centroids :

   Center1=Low{MidMark,LabMark,FinalMark}

   Center2=Medim{MidMark,LabMark,FinalMark}

   Center3=High{MidMark,LabMark,FinalMark}

2. From K- cluster by assigning all **Marks** to the closest centroids
3. Recomputed the centroid of each cluster.
4. **Results:**

We grouped the students regarding their final grades in three ways:

1. Assign possible labels that are the same as number of possible grades
2. Group the students in three classes “High”, “Medium”, “Low”.

3. Categorized the students with one of two class labels “Passed” for marks greater than or equal to 40 and “Failed” for marks less than 40

*Table 61: Clustering-Example 1*

| Class | Marks | No. of students | Percentage |
|---|---|---|---|
| A+ | 98-100 | 3 | 3% |
| A | 95-100 | 8 | 6% |
| A- | 90-95 | 3 | 3% |

*Table 62: Clustering-Example 2*

| CATEGORIES | NO.OF.STUDENTS |
|---|---|
| HIGH | 22 |
| MEDIUM | 12 |
| LOW | 4 |

*Table 63: Clustering-Example 3*

| Class | Marks | No. Of .Students | Percentage |
|---|---|---|---|
| **Passed** | 40< = Percentage | 28 | 73% |
| **Failed** | 40> Percentage | 10 | 26% |

## CONCLUSION:

Data mining allows the user to analyze data from different dimensions categorize it and summarize the relationship. Identify during mining process. Data mining techniques are used to operate on large volume of data to discover hidden pattern and relationship helpful in decision making. Different data mining techniques are used in the field of education. Cluster analysis used to segment a large set of data into subsets. Each cluster is collection of data objects that are similar to another placed within the same cluster but dissimilar to objects in other cluster. Clustering is one of the basic techniques often used in analyzing data sets. This study makes

use of cluster analysis to segment students into groups according to their characteristics.

## 2.1. Libraries:

- **DES Library:**

  Description: Represents the base class for the Data Encryption Standard (DES) algorithm from which all DES implementations must derive. Contain {DES Fields, DES Methods, DES Properties, DES Constructor.

  **Namespace:** System. Security. Cryptography.

  **Assembly:** mscorlib (in mscorlib.dll).

- **RSA Library:** Description: Represents the base class from which all implementations of the RSA algorithm inherit. Contain {RSA Fields, RSA Methods, RSA Properties, RSA Constructor.

  **Namespace:** System. Security. Cryptography

  **Assembly:** mscorlib (in mscorlib.dll)

- **NMath Library:** Description: The NMath .NET math library contains foundational classes for object-oriented numeric on the .NET platform.

  Namespace: Works with all .NET languages, includingC#, VisualBasicF#.

- **Enumerable Library:** Provides a set of static (Shared in Visual Basic) methods for querying objects that implement II Enumerable>.

  **Namespace:** System. LINQ .

  **Assembly:** System. Core (in System.Core.dll).

## 2.2. Tools:

- Visual Studio 2017.
- Fiddler-Free Web Debugging Proxy.
- Microsoft office 2016.
- Microsoft SQL Server.
- GitHub.
- Chrome Browser.
- Sublime Text.

# FURTHER ABROACH

In this chapter we present the main techniques and ways that we attempt to use in the future to improve EAS.

1. Improve speed.
2. Make IOS & Android Application.
3. Improve Security.
4. Create Endpoint for.
    a. Placement assessment.
    b. Formative assessment.
5. Improve Intelligence

# LIST OF FIGURES

# LIST OF TABLES

# REFERENCES


1. Allen, M.J. (2004). Assessing Academic Programs in Higher Education. San Francisco: Jossey-Bass.

2. Online Descriptive Examination and Assessment System.L. Zhang, et al., Development of Standard Examination System of Special Course for Remote Education, Journal of Donghua University (English Edition), 2013, Vol. 19, NO.1, 99-102.

3. Design Patterns Explained: A New Perspective on Object-Oriented Design (2nd Edition).

4. Black, P. J. ( 1993). Formative and summative assessment by teachers. Studies in Science Education, 21, 49-97.

5. Black, P., & Wiliam, D. ( 1998a). Assessment and classroom learning. Assessment in Education, 5(1), 7-74.

6. Cunningham, G. K. ( 1997). Assessment in the classroom. London, England

7. Hein, G., & Price, S. ( 1994). Active assessment for active science.Portsmouth, NH: Heinemann.

8. Shaeela Ayesha,(2010),data mining model for higher education system, J. of scientific research, vol -43, pp24-29.

9. . https://en.wikipedia.org/wiki/Main_Pageئ

10. https://docs.microsoft.com/en-us/aspnet/web-api

11. https://www.asp.net/mvc

12. https://stackoverflow.com

13. ASSESSMENT TYPES https://www.adelaide.edu.au/learning/teaching/assessment/introduction/assessmentTypes.pdf

# NOTES