

Yarmouk Private University Faculty of Informatics and Communication Engineering Department of Software Engineering Major

**Education Assessment System**

A project to gain the Bachelor degree in software engineering

**Developed By**

Seif Al-Deen Al-Kowatli Saleem Jammoul

**Supervised By**

Dr. Eng. Tala Shehbi

D.Eng. Wassim Juneidi

Semester

2017-2

1. Content …………………………………………………………………..… 2
2. Abstract …………………………………………………………...…….…. 4
3. Chapter 1: Introduction …………...………….….………….….………….. 5
   1. Education Assessment? ……………………………………………... 5
   2. Online Exam System Features ………………………….…………... 8
4. Chapter 2: History …………….………………………………………….. 10
   1. Background …………………….………………………………….. 10
5. Chapter 3: System Architecture ….………….….………….…….………. 18
   1. Database Server …….…………….…………….………….……… 20
      1. Entity Framework ………..…………………………………….. 20
      2. Database ERD ………..………………………………………… 22
   2. Web API Server …………………………………………………… 23
      1. Why to use ASP.NET Web API ……………………………….. 23
      2. Server & Client Sides in Web API ….…………………………. 25
      3. WCF VS. ASP.NET Web API ……….………………...……… 26
   3. ASP MVC Clients…….………………………………….……...… 28
      1. What is ASP MVC ….……………………………………….… 28
      2. ASP MVC Request ……..……………………………………... 29
      3. Technologies used in ASP MVC endpoints ….……………….. 30
6. Chapter 4: System Algorithms …………………………………………... 32
   1. Question Fetching Algorithm ………………………….…………... 32
   2. Auto Correction and auto Grading.……………………………….... 34
7. Chapter 5: Security ………………………………………………………. 36
   1. Denial of Service …………………………………………………... 36
   2. SQL Injection ……………………………………………………… 37
   3. Cross Site Scripting …………………………………….………….. 39
8. Chapter 6: System Diagrams.……………………………….….…….…… 40
   1. Use-Case Diagram …………………………………………..…….. 40
   2. Specifications ……….……………………………………………... 41
   3. Sequence diagrams………………………………………………… 65
   4. Collaboration Diagrams …………………………………………… 69
9. Chapter 7: Testing ……………………………………………….……….. 72
   1. Test Suits ………………………………………………………..…. 72
10. Conclusion …………………….………………………………………….. 74
11. References …………………….………………………………………….. 75
12. Index …….………………………………….…………………………….. 76

# 5. Chapter 3: System Architecture

To make the distribution of our server possible we used the Client-Server Model which is a distributed application structure that partitions tasks or workloads between the providers of a resource or services (Server) and service requesters (Clients). Often Clients and Servers communicate over a computer network on separate hardware.



**Figure 1: Client Server Model**

**Servers:**

* **Database Server**

The Database server connected to three Databases

1. **Question Bank DB**: Contains all questions of the system and all information about Exams and implemented in Oracle.
2. **Users DB**: Contains information about students and teacher and implemented in MS-SQL.
3. **Admin DB**: Contains information about admins and implemented in MySQL.

The Database server is handles all CRUD operation on all three Databases.

And to do the CRUD operation the server use Entity Framework Technology

* **API Server:**

The API Server Contains all the functionalities of our system and handle all client’s requests. The API server use ASP Web API technology.

* **Clients:**
  + ASP MVC Clients:

In this case the ASP server is a client to our API server. there are two types of those servers

1. The Examination Website server: Which is a website on the Intranet that Students Apply for the exam.
2. Student and Teacher Portal: this website allow student to browse their previous exams and show a detailed information about each exam, and also allow teachers to add, delete or edit questions of the question bank.

This a simple Diagram of how our servers and Clients Work together.

DB-Server

API

Server

ASP.NET

Server

Client

Client

Client

MVC Clients

**Figure 2: Online Exam Architecture**

# 5.1. Database Server

The Database server hold the three databases and preform all CRUD operations on all three databases to use perform this operation we used the Entity Framework.

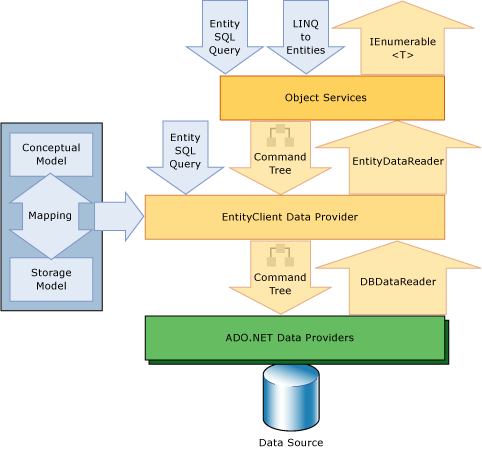
**5.1.1. Entity Framework**

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.

**Why Entity Framework?**



**Figure 3: Architecture of 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.



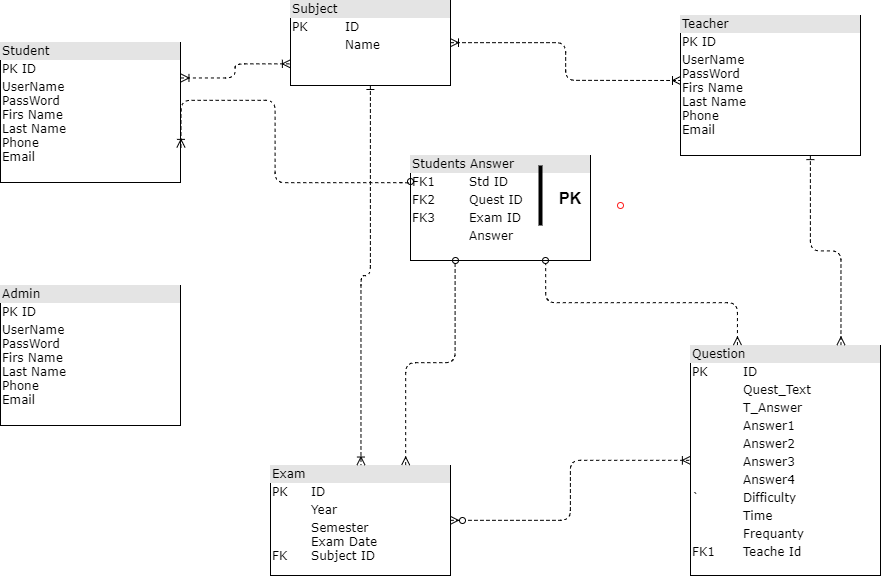
**Figure 4: ORM in Entity Framework**

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 5:The Mechanism of Entity Framework**

**5.1.2. Database ERD**



**Figure 6: Database ERD**

# 5.2. API Server

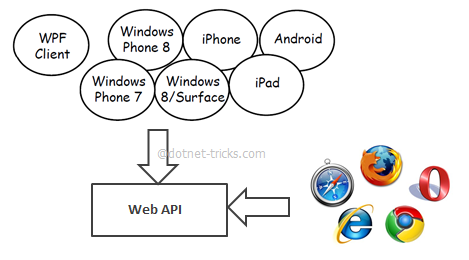
A **Web API** is an application programming interface (API) for either a web server or a web browser, usually limited to a web application's client-side (including any web frameworks being used). Asp.Net Web API is a framework for building HTTP services that can be consumed by a broad range of clients 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.

**5.2.1. 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 7:API Mechanism (Example)**

**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:**

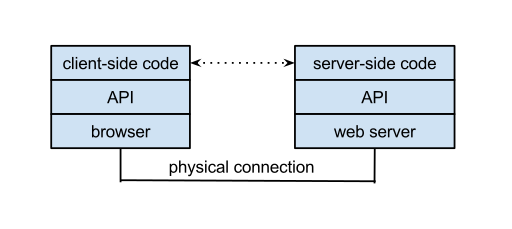
APIs available on both the client side and the server side:

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

**5.2.2. Client & Server in Web API:**

* **Server side:**

A server-side web API is a programmatic interface consisting of one or more publicly exposed endpoints to a defined request–response message system, typically expressed in JSON or XML, which is exposed via the web—most commonly by means of an HTTP-based web server. Mashups are web applications which combine the use of multiple server-side web APIs. Web hooks are server-side web APIs that take as input a Uniform Resource Identifier (URI) that is designed to be used like a remote named pipe or a type of callback such that the server acts as a client to dereference the provided URI and trigger an event on another server which handles this event thus providing a type of peer-to-peer IPC. There are some PHP microframeworks such as Lumen to build a REST API.



**Figure 8:Connection Between Client and Server in Web API**

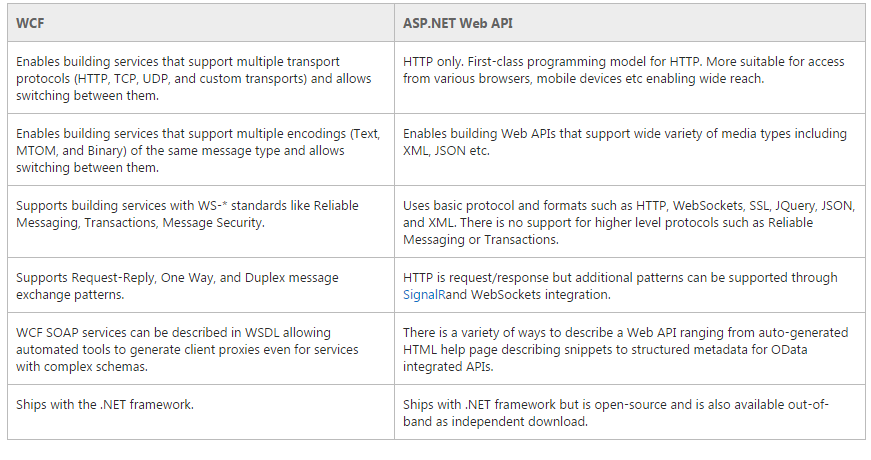
* **Client side:**

A client-side web API is a programmatic interface to extend functionality within a web browser or another HTTP client. Originally these were most commonly in the form of native plug-in browser extensions however most newer ones’ target standardized JavaScript bindings. The Mozilla Foundation created their WebAPI specification which is designed to help replace native mobile applications with HTML5 applications. Google created their Native Client architecture which is designed to help replace insecure native plug-ins with secure native sandboxed extensions and applications. They have also made this portable by employing a modified LLVM AOT compiler.

**5.2.3. 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:



**Figure 9:WCF VS. ASP.NET Web API**

**Why to choose Web API over WCF?**

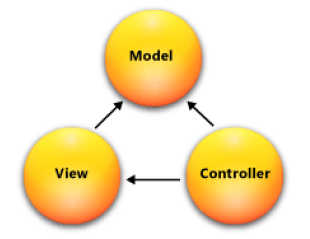
1. If we need a Web Service and don’t need SOAP, then ASP.Net Web API is best choice.
2. It is Used to build simple, non-SOAP-based HTTP Services on top of existing WCF message pipeline.
3. It doesn't have tedious and extensive configuration like WCF REST service.
4. Simple service creation with Web API. With WCF REST Services, service creation is difficult.
5. It is only based on HTTP and easy to define, expose and consume in a REST-
6. It is light weight architecture and good for devices which have limited bandwidth like smart phones.
7. It is open source

**API Server Job:**

The API server handles all HTTP requests of all clients this the types of requests that the API server handles.

1. Get Request: Retrieve data of certain object.
2. Post Request: create new instance of object.
3. Put Request: Modify the data of an object.
4. Delete Request: Delete certain object.

**5.3. ASP.NET MVC Clients**



**Figure 10: MVC Model**

**5.3.1. What is MVC?**

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

**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

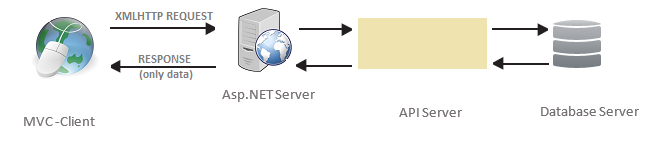
1. **Online Examination:** handles the online examination process and send requests through the intranet
2. **Online Examination Gate & Admin Panel:** consist of three main sections
3. **Students Gate:** Allow students to show their exams and requests detailed report about a certain exam.
4. **Teacher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

**5.3.2. 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:
   1. done (response, status, jqXHR) - called when the response from the server is successful
   2. fail (jqXHR, status, error) - called when the response from the server fails or the request times out.
   3. complete (response, status, jqXHR) - Always called when a response is received from the server



**Figure 11: How Ajax Request is Made**

**5.3.3. 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.

# 6.Chapter 4: Algorithms

**6.1. Questions Fetching Algorithm**

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

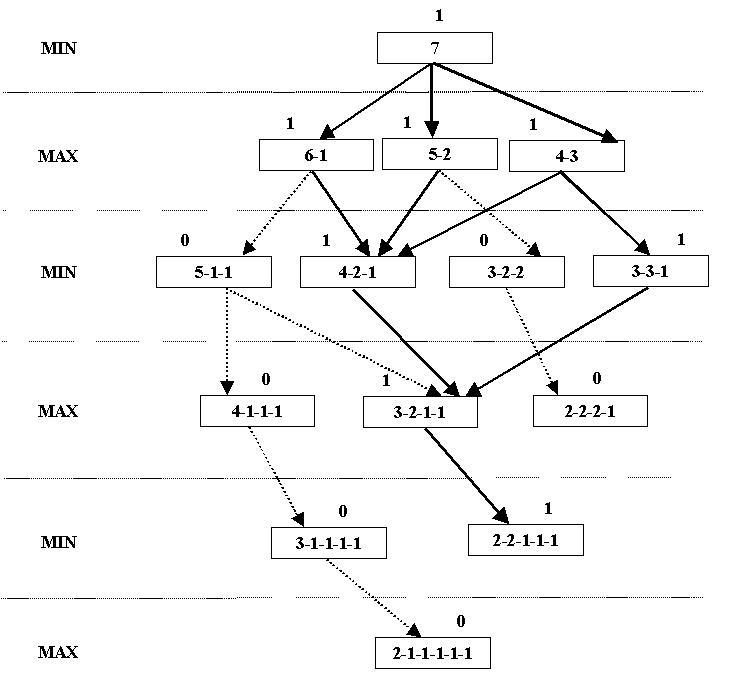
1. Degree of difficulty of the Questions pattern must be fair to all students.
2. The Questions pattern should be Different from one student to another. To ensure that we designed question algorithm using the Grundy’s Numbers.

**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 con- siting 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 12: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 13: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.

**6.2. Questions Correction**

One of the most important functionalities in our system is the auto correction and auto grading. There are two methods to do this functionality.

**First method:**

Operation of Questions correction starts at the end of the exam for each student. But after applying this method, a lot of problems appeared. And that’s because of the high demand for database connections:

1. Overlapping information.
2. System breakdown due to Accumulation of information.

**Second method:**

After each submission, Questions correction starts and save the result individually in the list, and When the exam time ends, the system calculates the final result and show the result directly to the student and store it in the database Correction operation.

we use second method in our examination system, so the process is done as follows:

1. The student enters his answer.
2. The system makes sure the answer is correct or not and stores the result.
3. At the end of time, the final result is calculated as follows let's have a table with student answers.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Questions | Mark | True answer | Student answer’s | result |
| Q1… | 3 | one | one | true |
| Q2… | 1 | tow | four | false |
| Q3… | 5 | three | ten | false |
| Q4…. | 2 | four | seven | false |
| Q5…... | 5 | Five | five | true |

Figure 14:Question Correction Algorithm Example

**Result** = (marks of true answer \*100) / all marks.

Calculate the result for the previous student:

**Result** = 3+ 5 \* 100 /16 =50%

**Success Factor** = Number of Answered Questions /Total Number of Questions

Final Result = Result /Success Factor.

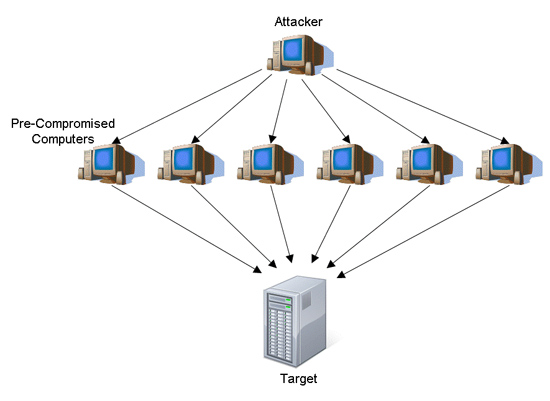
# 7.Chapter 5: Security

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

**7.1. 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 15: 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

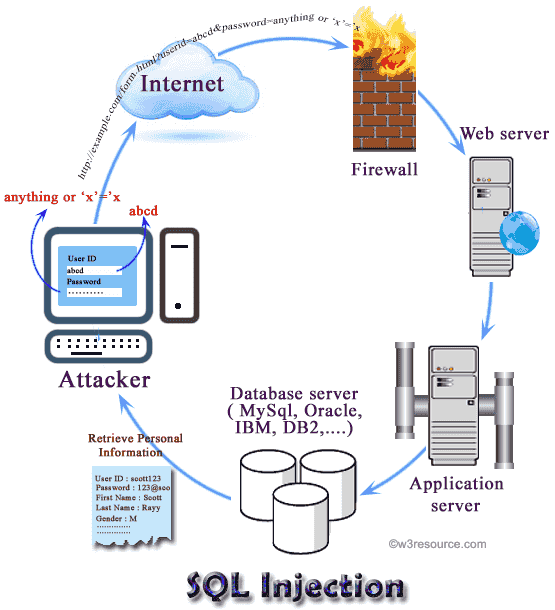
1. Identify the attackers
2. Block the response only to the attackers

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.

**7.2. SQL Injection**

SQL injection is a code injection technique, used to attack data-driven applications, in which nefarious SQL statements are inserted into an entry field for execution (e.g. to dump the database contents to the attacker). SQL injection must exploit a security vulnerability in an application's software, for example, when user input is either incorrectly filtered for string literal escape characters embedded in SQL statements or user input is not strongly typed and unexpectedly executed. SQL injection is mostly known as an attack vector for websites but can be used to attack any type of SQL database. SQL injection attacks allow attackers to spoof identity, tamper with existing data, cause repudiation issues such as voiding transactions or changing balances, allow the complete disclosure of all data on the system, destroy the data or make it otherwise unavailable, and become administrators of the database server.



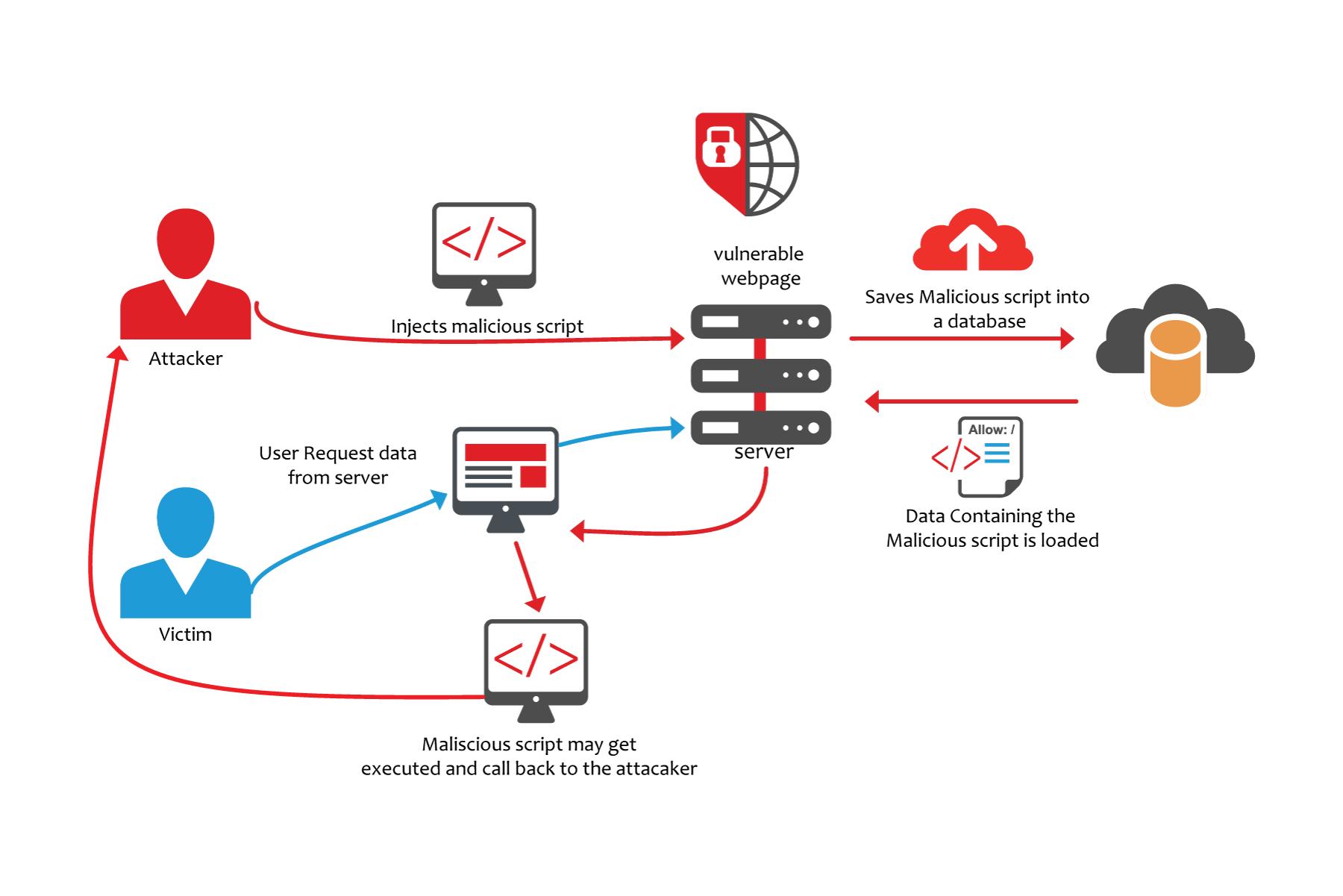
**Figure 16:SQL Injection attack**

**Counter measures against SQL-Injection:**

By using Entity Framework will issue parameterized SQL statements when an operation is executed. Using them will provide protection against SQL Injection attack without an extra effort. Using this data access mechanism can also prevent you from trouble because you can just program against the conceptual application model instead of programming directly against your database. You don’t have to deal with those typo and SQL syntax.

**7.3. Cross-site scripting**

attacks use known vulnerabilities in web-based applications, their servers, or the plug-in systems on which they rely. Exploiting one of these, attackers fold malicious content into the content being delivered from the compromise zed site. When the resulting combined content arrives at the client-side web browser, it has all been delivered from the trusted source, and thus operates under the permissions granted to that system. By finding ways of injecting malicious scripts into web pages, an attacker can gain elevated access-privileges to sensitive page content, to session cookies, and to a variety of other information maintained by the browser on behalf of the user. Cross-site scripting attacks are a case of code injection



**Figure 17:XSS attack**

**Counter measures against XSS:**

to prevent XSS attacks we used AntiXSS which is an encoding library, designed and developed by CISG team at Microsoft in conjunction with the ACE Team. It is designed to help developers protect their Web-based applications from XSS attacks. This library is very different from most encoding libraries; it uses the principle-of-inclusions technique to provide protection against XSS attacks. This approach works by defining a valid or allowable set of characters, and encoding anything outside this set (invalid characters or potential attacks). It offers several advantages over other encoding schemes.