Zagazig University

Faculty of Computer and Informatics

Department of Decision Support

(2022_2023)

B. Eng. Final Year Project

# Youth Care Department Automation

By:

Ahmed Adel Elsaid Gad

AbdElhamid Mohamed Elbakry

Marwan Atif Ahmed Kamal

Mostafa Ehab Mostafa Khalil

Ahmed Essam Elharty

Abdullah Mahmoud Attia

Mohamed Eid AboEldhab

Supervised By:

Dr/ Shimaa Saber

Eng/ Esraa Kamal

# ABSTRACT

The aim of this graduation project is to develop a website that caters to the needs of the Youth Care Department at the College of Computer and Information Sciences. The website is designed to provide a one-stop platform for students to access information and services related to their well-being and development.

The website will include sections dedicated to various committees within the department, such as the Sports Committee, Technical Committee, Mobile Committee, Public Service Committee, Social and Trips Committee, Student Family Committee, Science and Technology Committee, and Cultural Activity Committee .

Each section of the website will provide information and resources related to the specific committee's objectives. For example, the Sports Committee section will include information on the sports activities available on campus, while the Technical Committee section will provide resources and tutorials on various technical skills and tools. The website's overall goal is to provide a centralized platform for students to access information and resources related to their personal, social, and academic development .

The website for the Youth Care Department at the College of Computer and Information Sciences will have a user-friendly interface that is easy to navigate.

The homepage will serve as a gateway to the various sections of the website

Overall, the website will serve as an important resource for students in the Youth Care Department at the College of Computer and Information Sciences, by providing them with the tools and resources they need to succeed academically, professionally, and personally.

# Table of Contents

# Chapter one

## Introduction

# 1.1Background and Motivation

The background and motivation for the graduation project of developing a website for the Youth Care Department at the College of Computer and Information Sciences stem from the growing need for a centralized platform that provides information and resources related to the personal, social, and academic development of students.

The Youth Care Department is responsible for providing support and care for students, and the college recognizes the importance of addressing students' needs beyond their academic requirements. The department comprises several committees, each with a distinct focus, such as sports, technical skills, social activities, and more.

However, currently, there is no centralized platform for students to access information on the activities and resources provided by these committees. Students may need to navigate through various platforms, such as social media pages, posters, or notice boards, to gather information on the events and activities offered by the department. This can be time-consuming and confusing, leading to a lack of participation from students.

Therefore, the motivation behind this project is to develop a specialized website that provides a one-stop platform for students to access information and resources related to the Youth Care Department's activities and services. The website will make it easier for students to learn about the various committees and their objectives, and provide information on upcoming events and initiatives. This website will encourage students to participate in the department's activities and services, leading to a more supportive and engaging environment for students.

Overall, the project's background and motivation are rooted in the desire to provide a centralized platform that supports students' personal, social, and academic development, and enhances their overall college experience.

# 1.2Problem Definition

The project aims to develop a website for the Youth Care Department at the College of Computer and Information Sciences. The department is in need of a platform that effectively showcases the benefits of the department to potential students while also providing assistance to those who cannot afford the tuition fees.

The current problem is that the department lacks a comprehensive and user-friendly website that can attract and inform potential students about the various services offered, including counseling, mentorship, and extracurricular activities. Additionally, there is a need for a platform that can connect students who are facing financial difficulties with available financial aid programs and scholarships.

# 1.3- Purpose of the Project:

The website for the Youth Care Department at the College of Computer and Information Sciences will be designed to be user-friendly and visually appealing, with easy navigation and clear calls-to-action. It will provide an overview of the department's services, including counseling, mentorship, extracurricular activities, and community outreach programs.

One of the key features of the website will be a section dedicated to financial aid and scholarships. This section will provide information on available programs and scholarships, eligibility requirements, and application deadlines. Students who are facing financial difficulties will be able to access this information easily and get the support they need to continue their education.

Overall, the goal of the project is to create a website that effectively communicates the value of the Youth Care Department at the College of Computer and Information Sciences and provides students with the resources they need to succeed. By doing so, the website will help improve the overall student experience at the college and contribute to the success of its students.

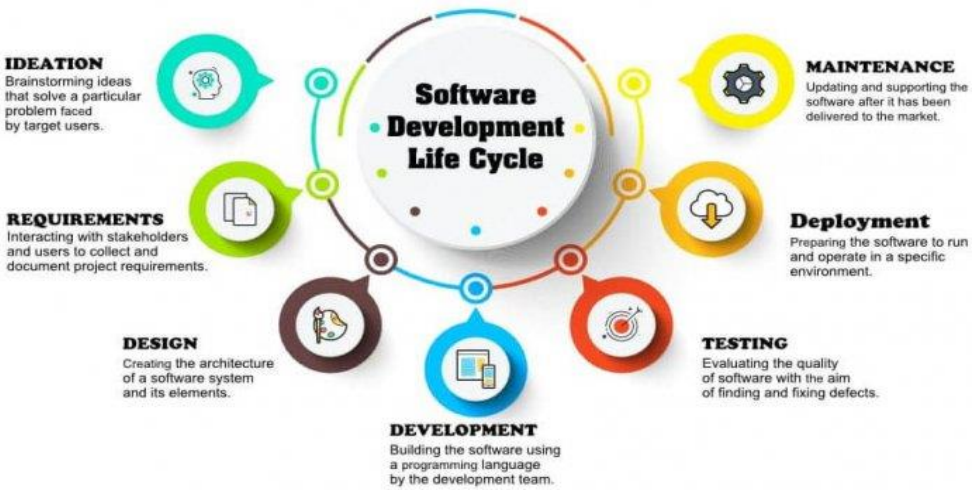
# Chapter two

## System Analysis

# 2.1- Introduction

- Systems analysis is a process of collecting factual data, understand the processes involved, identifying problems and recommending feasible suggestions for improving the system functioning.
  This involves studying the business processes, gathering operational data, understand the information flow, finding out bottlenecks and evolving solutions for overcomingthe weaknesses of the system so as to achieve the organizational goals.
  System Analysis also includes subdividing of complex process involving the entire system, identification of data storeand manual processes.
  The major objectives of systems analysis are to find answers for each business process: What is being done, how is it being done, who is doing it, when is he doing it,why is it being done and How can it be improved? It is more of a thinking process and involves the creative skills of the System Analyst.
  It attempts to give birth to a new efficient system that satisfies the current needs of the user and has scope for future growth within the organizational constraints. The result of this process is a logical system design.
  Systems analysis is an iterativeprocess that continues until a preferred and acceptable solution emerges.

- The systems development life cycle (SDLC) is the process of determining how an information system (IS) can support business needs, designing the system building it, and delivering it to users. It helps in producing a software with the highest quality and lowest cost in the shortest time, SDLC includes a detailed plan for how to develop, alter maintain, and replace a software system.




## ➢ Planning phase

The planning phase is the fundamental process of understanding why an information system should be built and determining how the project team will go about building it.

- Why an information system should be built?
- How will the project team go about building it?

## ➢ Analysis Phase

During this phase, the project team investigates any current system(s), identifies improvement opportunities, and develops a concept for the new system. The analysis phase answers the questions of who will use the system, what the system will do, and where and when it will be used.

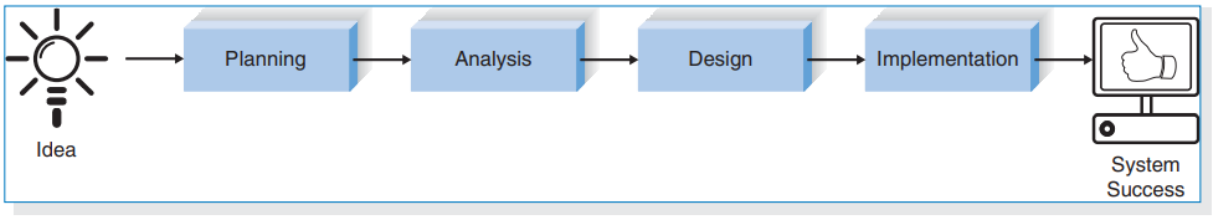
## ➢ Design Phase

The design phase decides how the system will operate, in terms of the hardware, software, and network infrastructure; the user interface, forms, and reports that will be used; and the specific programs, databases, and files that will be needed.

## ➢ Implementation Phase

During the implementation phase, the system is either developed or purchased (in the case of packaged software) and installed.

This phase is usually the longest and most expensive part of the process.



# 2.2- Planning Phase

## 2.2.1- **Project Definition**

The Youth Care Department at the College of Computer and Information Sciences (CCIS) is responsible for providing a range of services and support to students to promote their personal and academic development.

The department currently operates using a manual system, which can be time-consuming and inefficient. As part of our graduation project, we will be designing and developing a website for the Youth Care Department to streamline its operations and enhance its services.

In this chapter, we will conduct a system analysis to identify the requirements and design of the website.

# 2.3- Analysis Phase

## 2.3.1- Project Methodology

The current manual system used by the Youth Care Department is inefficient and can lead to delays in providing support to students. The lack of an online platform also makes it difficult for students to access information and resources related to the services provided by the department. The problem, therefore, is to design and develop a website that will automate the operations of the Youth Care Department and provide easy access to information and resources for students.

## 2.3.2 – Requirements

The system requirements for the Youth Care Department website can be divided into functional and non-functional requirements. Functional requirements include the features and functions that the website must provide to meet the needs of the department and its stakeholders. These include:

### 2.3.2.1- Functional Requirements:

- An online platform for students to request support and services
- A database to store student information and service requests
- A dashboard for department staff to manage and process service requests
- A calendar for scheduling appointments and events
- A library of resources for students, including articles, videos, and webinars
- A feedback system for students to provide input on the quality of services provided
- A news section to inform students about upcoming events and activities

### 2.3.2.2- Nonfunctional Requirements:

Non-functional requirements refer to the quality attributes that the system must possess, such as performance, usability, and security. These include:

- A user-friendly interface that is easy to navigate
- Fast response time and minimal downtime
- Strong security measures to protect student information and data
- Compatibility with different devices and web browsers

## 2.3.3- Use Case

A use case is a methodology used in system analysis to identify, clarify, and organize system requirements. The use case is made up of a set of possible sequences of interactions between systems and users in a particular environment and related to a particular goal. It consists of a group of elements (for example, classes and interfaces) that can be used together in a way that will have an effect larger.

In system analysis, a use case is a description of a specific interaction between a user and a system that results in a measurable outcome. A use case describes the steps that a user takes to achieve a specific goal within the system.

Use cases are an important tool in system analysis because they help to identify the functional requirements of the system. By analyzing use cases, system analysts can identify the specific features and functionality that the system must provide in order to meet the needs of its users.

A typical use case includes the following elements:

- **Actor**: the person or system that initiates the use case
- **Goal**: the objective that the actor is trying to achieve
- **Preconditions**: the conditions that must be met before the use case can begin
- **Steps**: the sequence of steps that the actor takes to achieve the goal
- **Postconditions**: the state of the system after the use case is completed
- **Exceptions**: any errors or exceptions that can occur during the use case

By analyzing use cases, system analysts can gain a better understanding of the requirements of the system, which can help to inform the design and development process. Use cases can also be used to validate the system once it has been developed, by testing whether it meets the goals and requirements described in the use cases.

## Use Case:

Is a list of actions or event steps, typically defining the interaction between a role (known in the Unified Modeling Language as an actor) and a system, to achieve a goal.

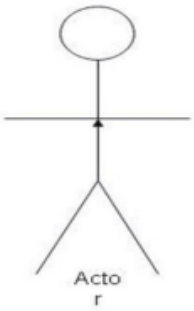
## The actor:

Actor might be a person, a company or organization, a computer program, or a computer system—hardware, software, or both. Actor categories:

1-Actor: anything with behavior that acts on the system.

2-Primary actor: initiates interaction to achieve goal (when system is a software product, primary actor is often the computer user).

3-Supporting actor: performs sub-goals to help use case.




## Use Case Relationship :

Three relationships among use cases are used often in practice.

## Include :

In one form of interaction, a given use case may include another. "Include is a Directed cases, implying that the inserted into the behavior of the including use case" The first use case often depends on the outcome of the included use case.

This is useful for extracting truly common behaviors from multiple use cases into asingle the notation is a dashed arrow from the including to the included use case, with the label "«include»". This usage resembles a macro expansion where the included use case behavior is placed inline in the base use case behavior.

There are no parameters or return values. To specify the location in a flow of events in which the base use case includes the Relationship between two usebehaviors of the included use case is description.

Behavior of another, you simply write include followed by the name of use caseyou want to include, as in the following flow for track order.

## Extend :

In another form of interaction, a given use case (the extension) may extend another. This relationship indicates that the behavior of the extension use case may be inserted in the extended use case under some conditions. The notation is a dashed arrow from the extension to the extended use case, with the label "«extend»".

Notes or constraints may be associated with this relationship to illustrate theconditions under which this behavior will be executed. Modelers use the

«extend» relationship to indicate use cases that are "optional" to the base use case. Depending on the modeler's approach "optional" may mean "potentially notexecuted with the base use case" or it may mean "not required to achieve the base use case goal.

## Generalization :

In the third form of relationship among use cases, a generalization / specializationrelationship exists. A given use case may be a specialized form of an existing usecase.

The notation is a solid line ending in a hollow triangle drawn from the specialized to the more general use case. This resembles the object-oriented concept of sub- classing, in practice it can be both useful and effective to factor out common behaviors, constraints and assumptions to the general use case, describe them once,and deal with it in the same way, except for the details in the specialized cases.

## System boundary boxes :

You can draw a rectangle around the Use case, called the system boundary box, to indicate the scope of your system. Anything within the box represents functionalitythat is in scope and anything outside the box is not. System boundary boxes are rarely used, although on occasion I have used them to identify which Use case willbe delivered in each major release of a system.

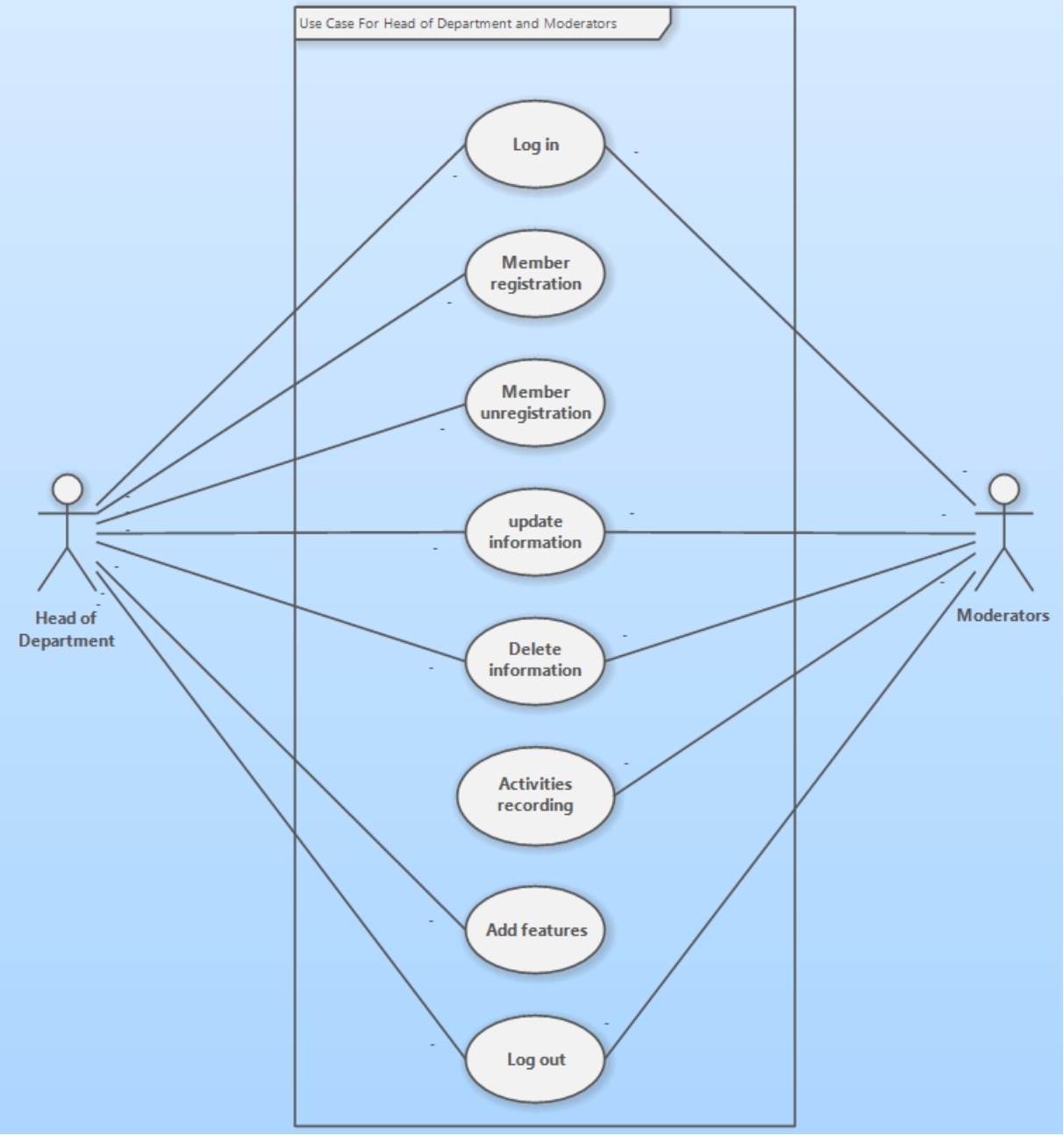
## Creating Use Case Diagram :

Start by identifying as many actors as possible. You should ask how the actors interact with the system to identify an initial set of Use case. Then, on the diagram,you connect the actors with the Use case with which they are involved. If actor supplies information, initiates the Use case, or receives any information as a result of the Use case, then there should be an arrowhead on the association lines becausemy experience is that people confuse them for indications of information flow, not initial Invocation.

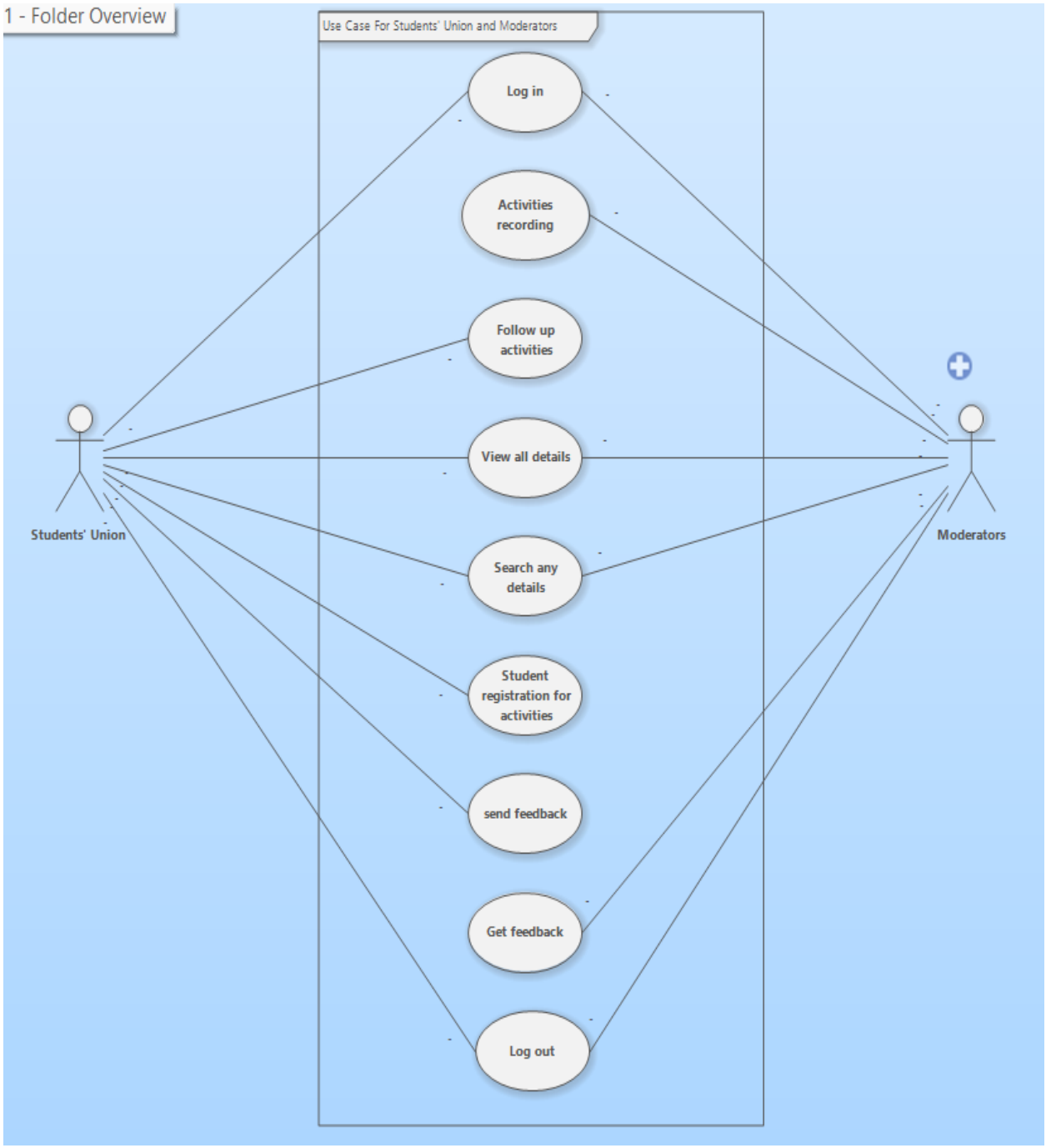
As l begins to notice similarities between Use cases, or between actors, I startmodeling the appropriate relationships between them.

## General Use Case for system :

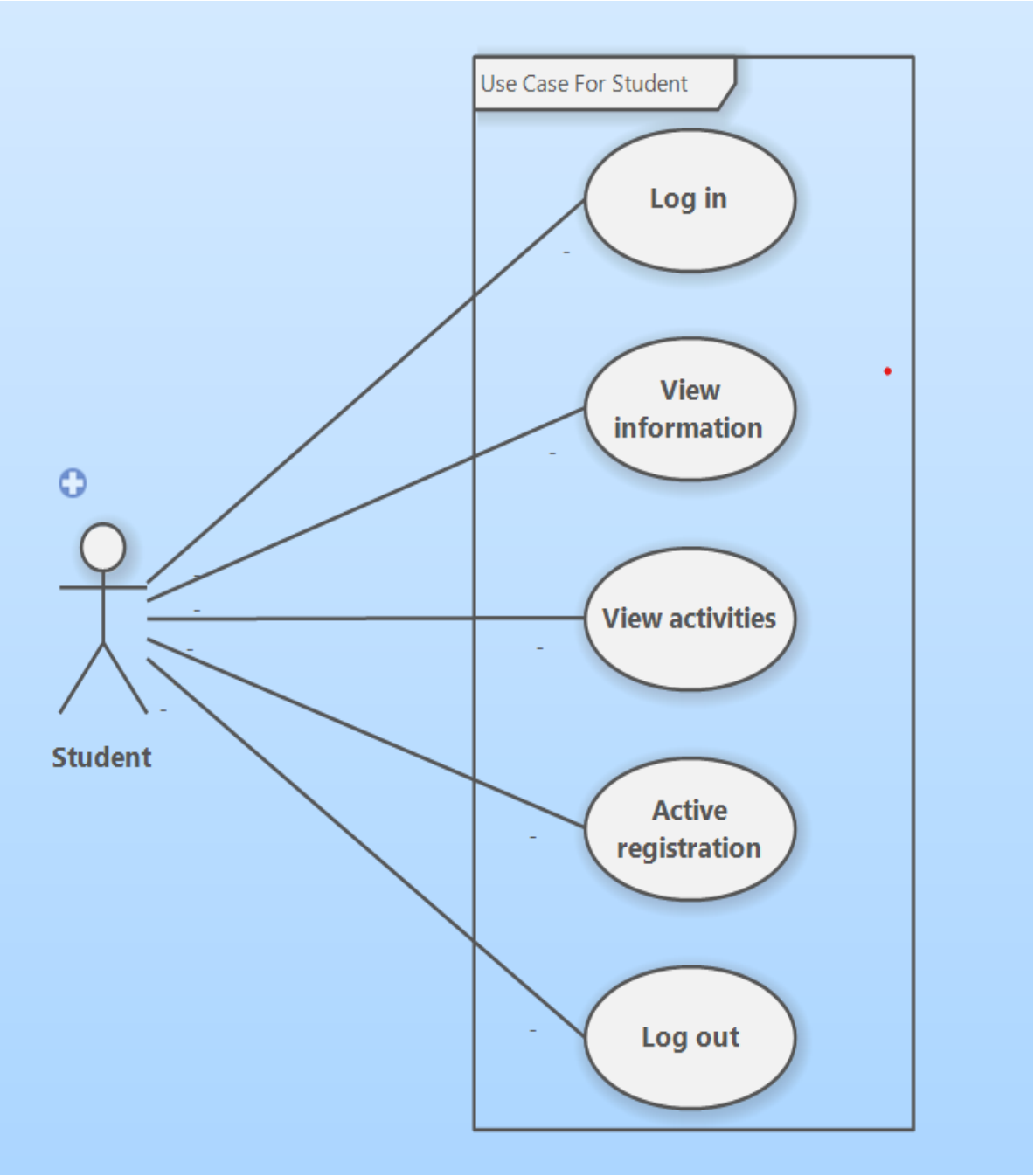
### Use Case For Head of Department and Moderators:



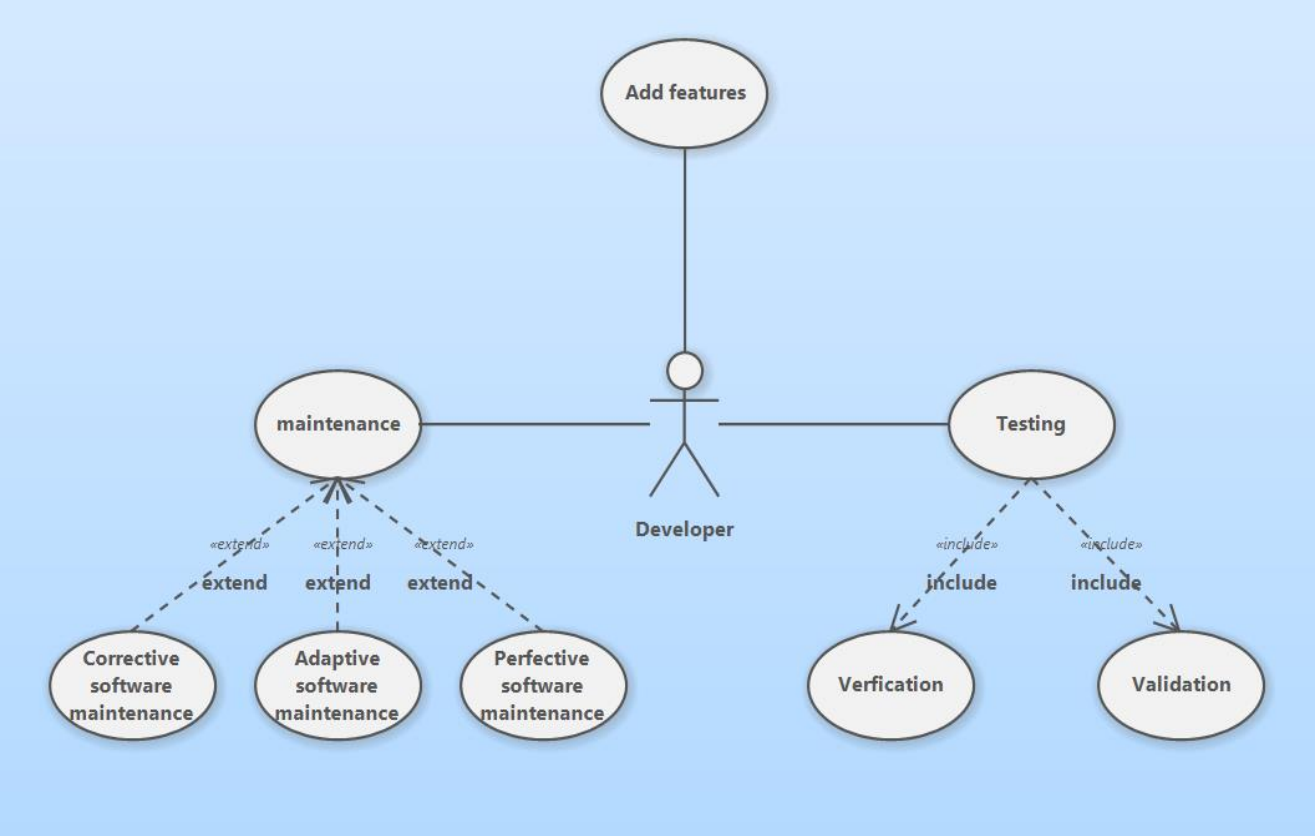
## Use Case For Students' Union and Moderators



## Use Case For Student:



# General Use Case for Developer:



## 2.3.4- Sequence Diagram

A sequence diagram is an interaction diagram that shows how objects operate withone another and in what order. It is constructing of a message sequence chart a sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called event diagrams or event scenarios.

A sequence diagram shows, as parallel vertical lines (lifelines), different processesor objects that live simultaneously, and, as horizontal arrows, the messages exchanged between them, in the order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner.

### 2.3.4.1- Sequence Diagram Notations

#### Class Roles or Participants:

Class roles describe the way an object will behave in context. Use theUML object symbol to illustrate class roles, but don't list object attributes

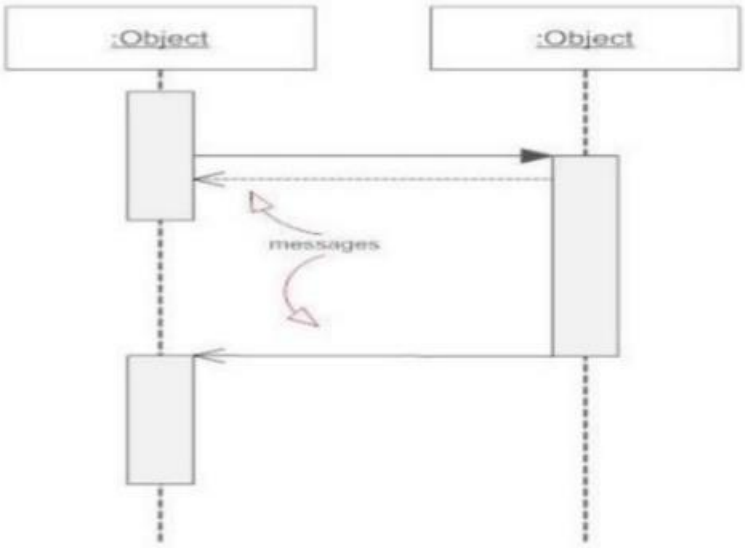
## Activation or Execution Occurrence :

Activation boxes represent the time an object needs to complete a task. When an object is busy executing a process or waiting for a reply message, use a thin gray rectangle placed vertically on its lifeline.


## Messages:

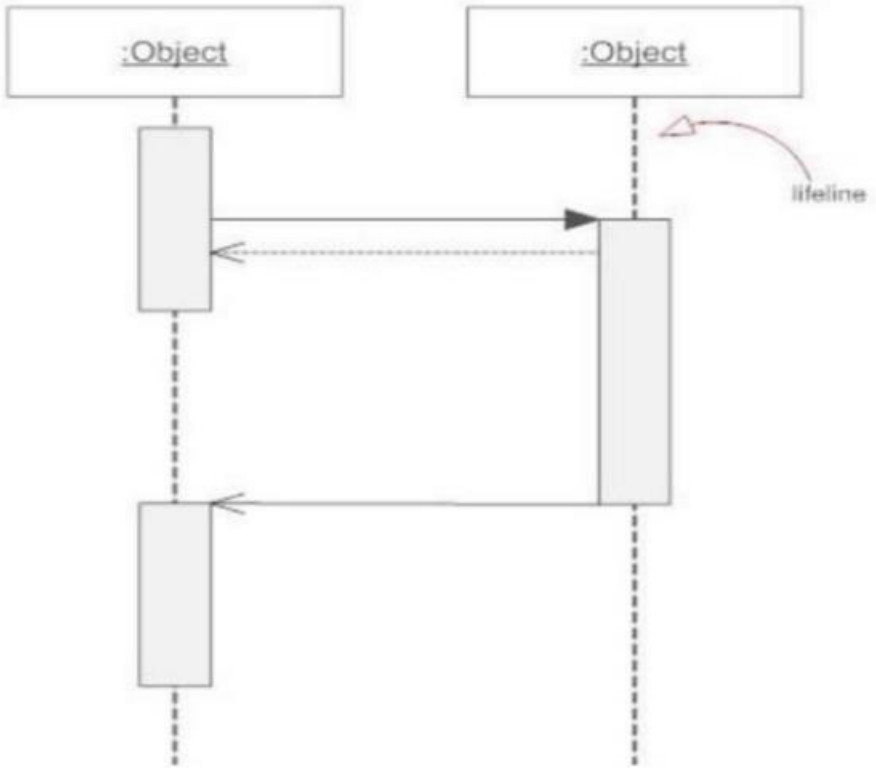
Messages are arrows that represent communication between objects. Use halfarrowed lines to represent asynchronous messages.

Asynchronous messages are sent from an object that will not wait for a response from the receiver before continuing its tasks. For message types, see below.



## Lifelines:

Lifelines are vertical dashed lines that indicate the object's presence over time.




## Destroying Objects:

Objects can be terminated early using an arrow labeled "<>" that points to an X. This object is removed from memory. When that object's lifeline ends, you can place an X at the end of its lifeline to denote a destruction occurrence.

## Loops:

A repetition or loop within a sequence diagram is depicted as a rectangle. Place the condition for exiting the loop at the bottom left corner in square brackets [ ].

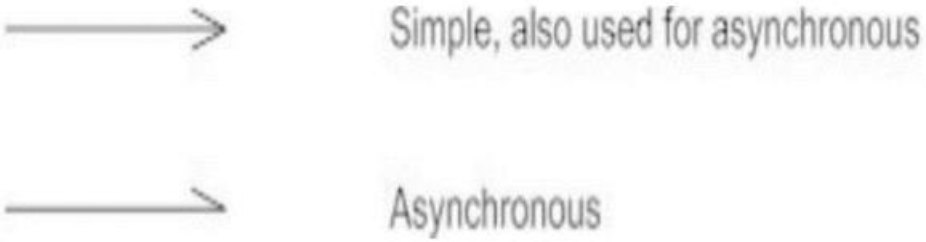
## 2.3.4.2- Message Types in Sequence Diagrams:

### Synchronous Message :

A synchronous message requires a response before the interaction can continue. It's usually drawn using a line with a solid arrowhead pointingfrom one object to another.


### Asynchronous Message :

Asynchronous messages don't need a reply for interaction to continue.Like synchronous messages, they are drawn with an arrow connecting two lifelines; however, the arrowhead is usually open and there's no return message depicted.




### Reply or Return Message :

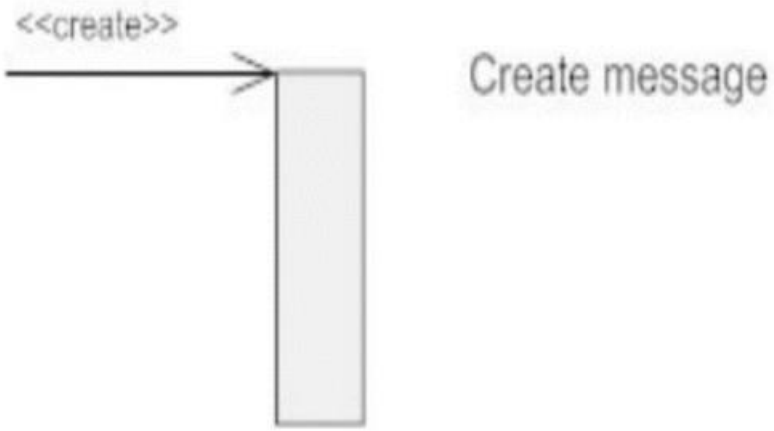
A reply message is drawn with a dotted line and an open arrowhead pointing back to the original lifeline

## Self-Message :

A message an object sends to itself, usually shown as a U shaped arrow pointing back to itself.


## Create Message:

This is a message that creates a new object. Similar to a return message, it's depicted with a dashed line and an open arrowhead that points to the rectangle representing the object created.



## Delete Message :

This is a message that destroys an object. It can be shown by an arrow with an x at the end.


## Found Message:

A message sent from an unknown recipient, shown by an arrow from an endpoint to a lifeline.


## Lost Message:

A message sent to an unknown recipient. It's shown by an arrow going from a lifeline to an endpoint, a filled circle or an x.


## Sequence diagram benefits:

1- You can easily see how tasks are distributed between components.
2- You can identify patterns of interaction that make it difficult to update the software.

## Creating sequence diagram:

As we have already discussed, the purpose of sequence diagrams is to capture the dynamic aspect of a system. So to capture the dynamic aspect, we need to understand what a dynamic aspect is and how it is visualized. Dynamic aspect can be defined as the snapshot of 28 the running system at a particular moment. The sequence diagram captures the time sequence of the message flow.

**From one object to another. Following things are to be identified clearly before drawing the sequence diagram:**

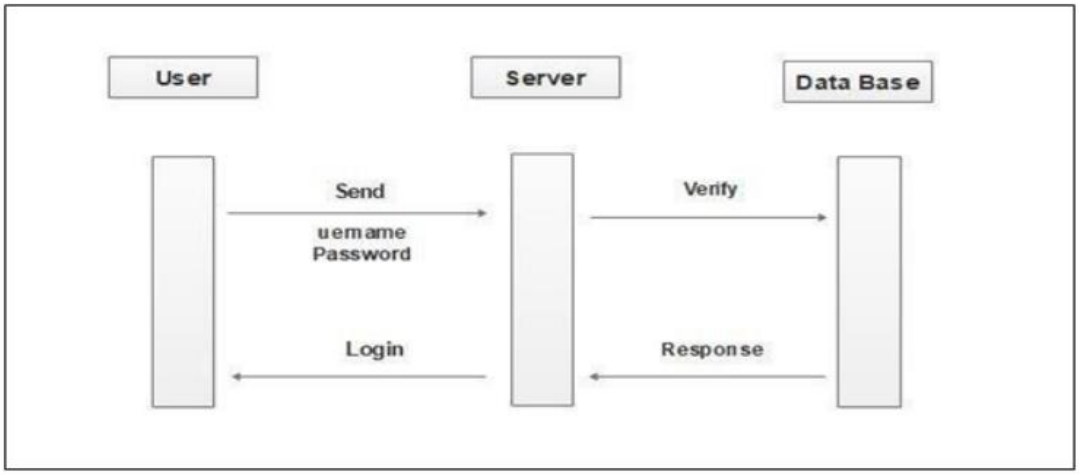
1. Objects taking part in the interaction.
2. Message flows among the objects.
3. The sequence in which the messages are flowing.
4. Object organization.

## Drawing Sequence Diagram :

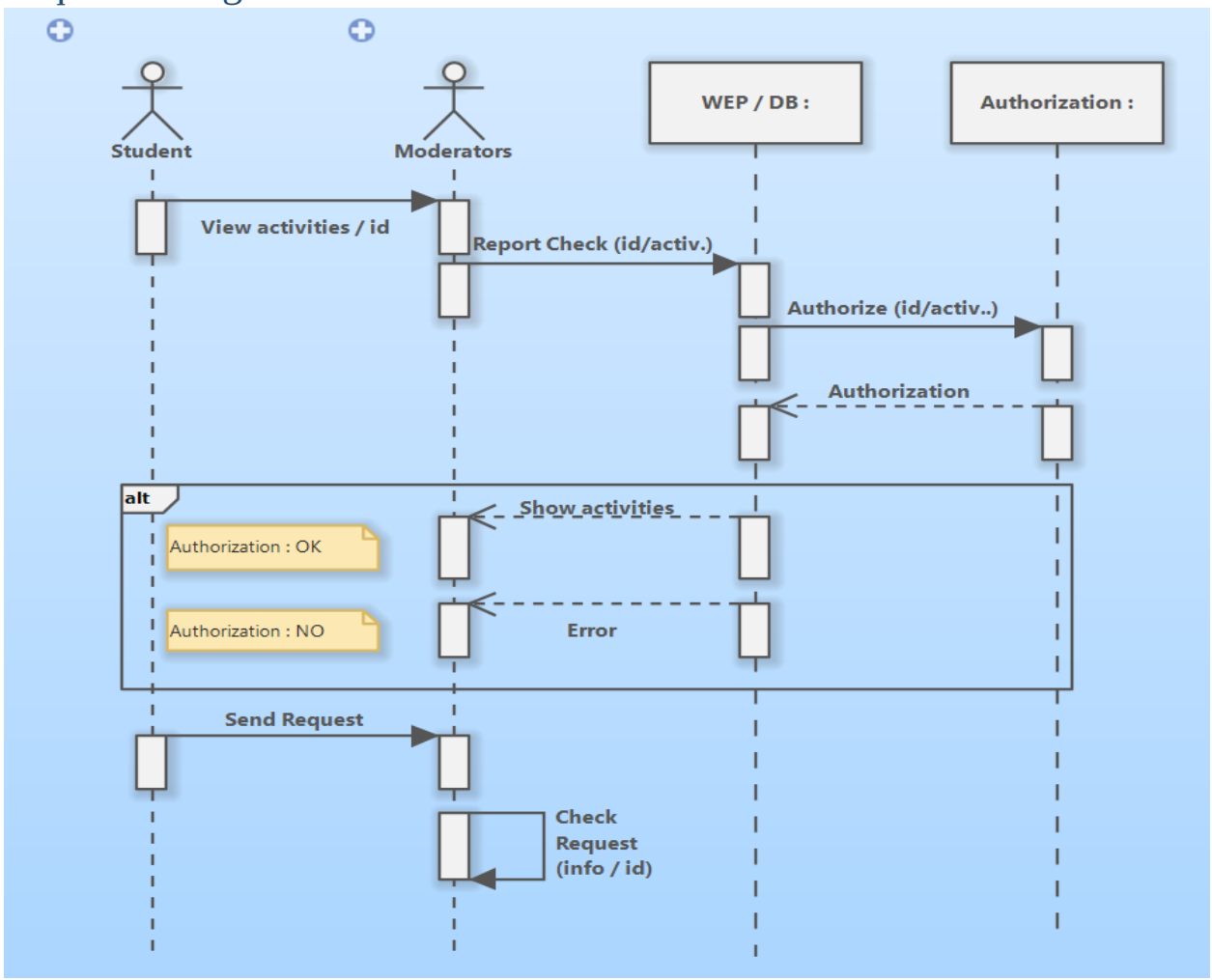
The sequence diagram captures the time sequence of the message flow from one object to another. Following things are to be identified clearly before drawing the interaction diagram:

- Objects taking part in the interaction.
- Message flows among the objects.

## Sequence diagram For login:




## Sequence diagram for Student and Moderators :



## 2.3.5- Class Diagram

In the Unified Modeling Language (UML) is a type of static structure diagram thatdescribes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects.

The class diagram is the main building block of object-oriented modeling. It is used both for general conceptual modeling of the systematic of the application, andfor detailed modeling translating the models into programming code. Class diagrams can also be used for data modeling.

The classes in a class diagram represent both the main elements, interactions in theapplicant, and the classes to be programmed.

## 2.3.5.1- Relationships

A relationship is a general term covering the specific types of logical connections found on class and objects diagrams. UML defines the following relationships:

### 1- Dependency :

A dependency is a semantic connection between dependent and independent model elements. It exists between two elements if changes to the definition of one element (the server or target) may cause changes to the other (the client or source). This association is uni directional.

### 2- Association :

An association represents a family of links. A binary association (with two ends) is normally represented as a line. An association can link any number of classes. An association with three links is called a ternary association. An association can be named, and the ends of an association can be adorned with role names, ownership indicators, multiplicity, visibility, and other properties.

### 3- Aggregation :

Aggregation is a variant of the "has an" association relationship; Aggregation is more specific than association. It is an association that represents a part-whole or part-of relationship. As shown in the image, a Professor 'has a' class to teach. As a type of association, an aggregation can be named and have the same adornments that an association can.

However, an aggregation may not involve more than two classes; it must be a binary Association.

During implementation, and the diagram may skip aggregation relations altogether. Aggregation can occur when a class is a collection or container of other classes, but the contained classes do not have a strong Lifecycle dependency on the container. The contents of the Container still exist when the container is destroyed.

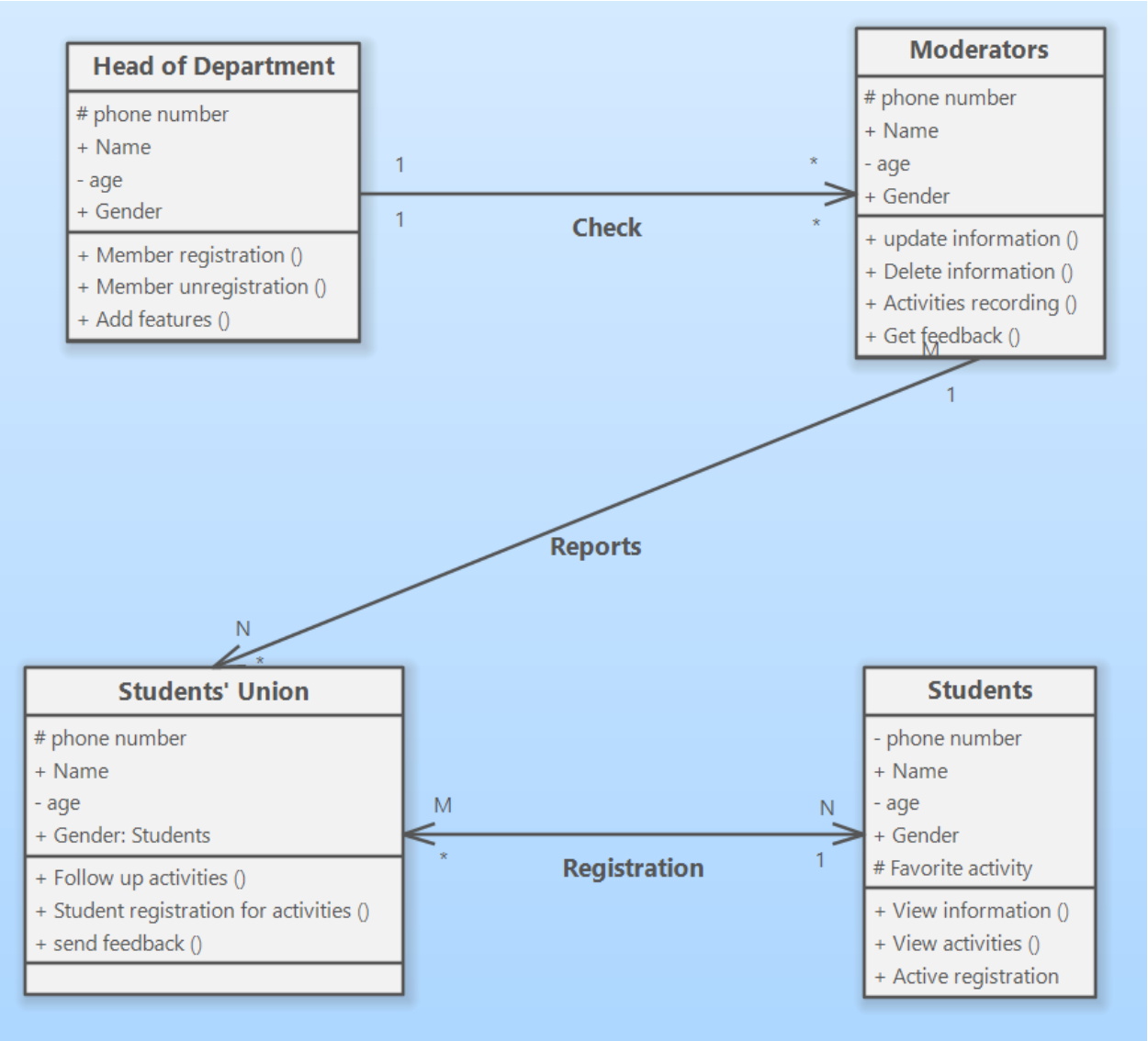
### 4- Composition :

The UML graphical representation of a composition relationship shows composition as a filled diamond shape on the containing class end of the lines that connect contained class to the containing class.

## 2.3.5.2- Purpose of class diagram

The purpose of class diagram is to model the static view of an website. Class diagrams are the only diagrams which can be directly mapped with object-oriented languages and thus widely used at the time of construction. UML diagrams like activity diagram, sequence diagram can only give the sequence flow of the website, however class diagram is a bit different. It is the most popular UML diagram in the coder community.

# Class Diagram For system:



# Conclusion

In conclusion, the system analysis for the Youth Care Department website is a critical component of our graduation project. By identifying the requirements and design of the website, we can ensure that it meets the needs of the department and its stakeholders and provides a streamlined and efficient platform for delivering support and services to students. In the next chapter, we will discuss the system design and development of the website.

# Chapter three

## Project design

## 3.1- Introduction

- In this chapter of the book we will identify the screen in the program and how to create it and the function of it.

- The design phase decides how the system will operate, in terms of the hardware, software, and network infrastructure; the user interface, forms, and reports that will be used; and the specific programs, databases, and files that will be needed.

**The design phase has four steps:**

1. Design Strategy: This clarifies whether the system will be developed by the company or outside the company.

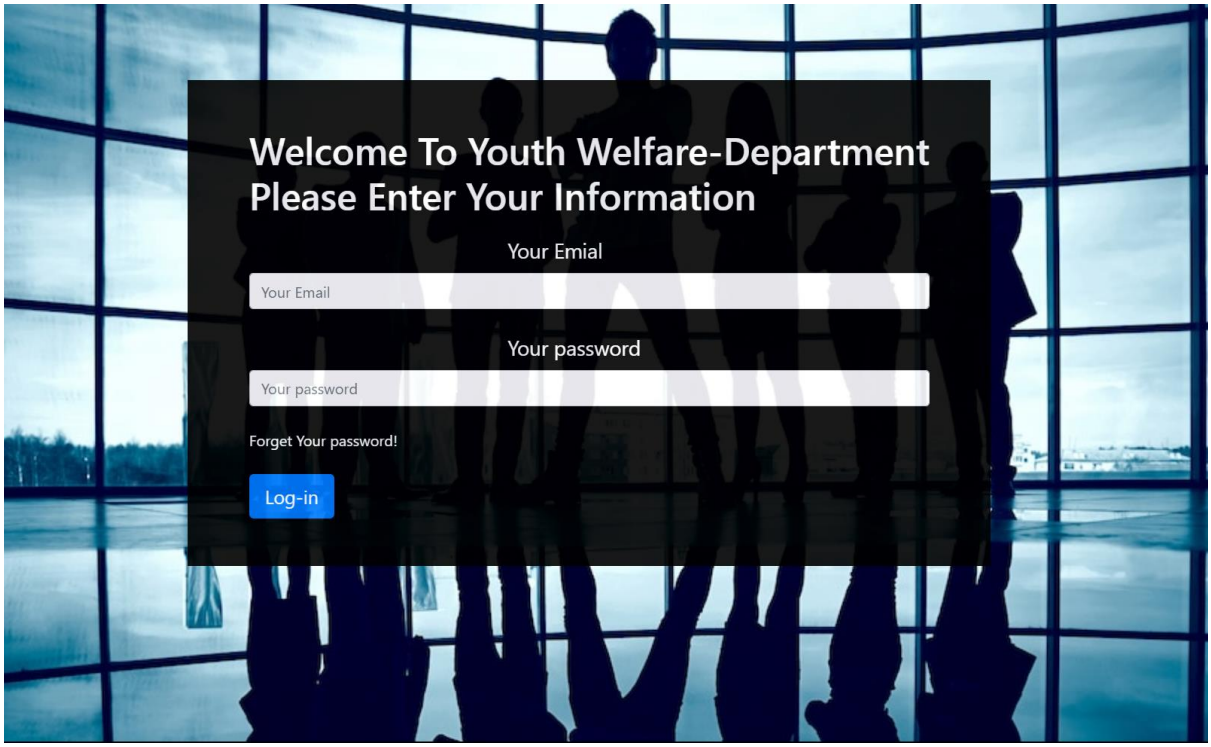
2. Architecture Design: This describes the hardware, software, and network infrastructure that will be used.

3. Database and File Specifications: These documents define what and where the data will be stored.

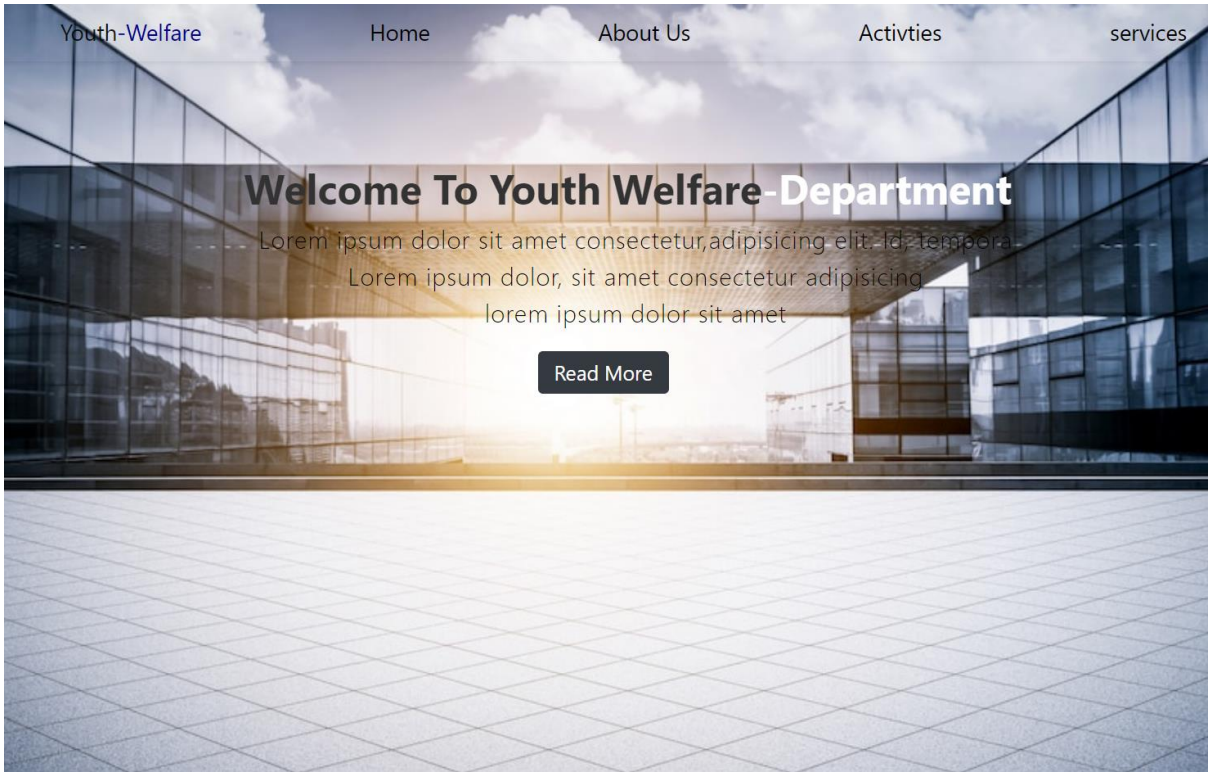
4. Program Design: Defines what programs need to be written and what they will do.

## 3.2.- User Interface & UI

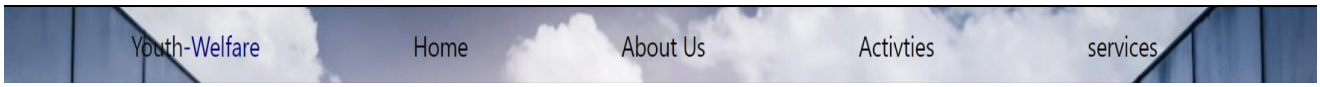
### 1.Login as User Form :




### 2.First page of The Website:

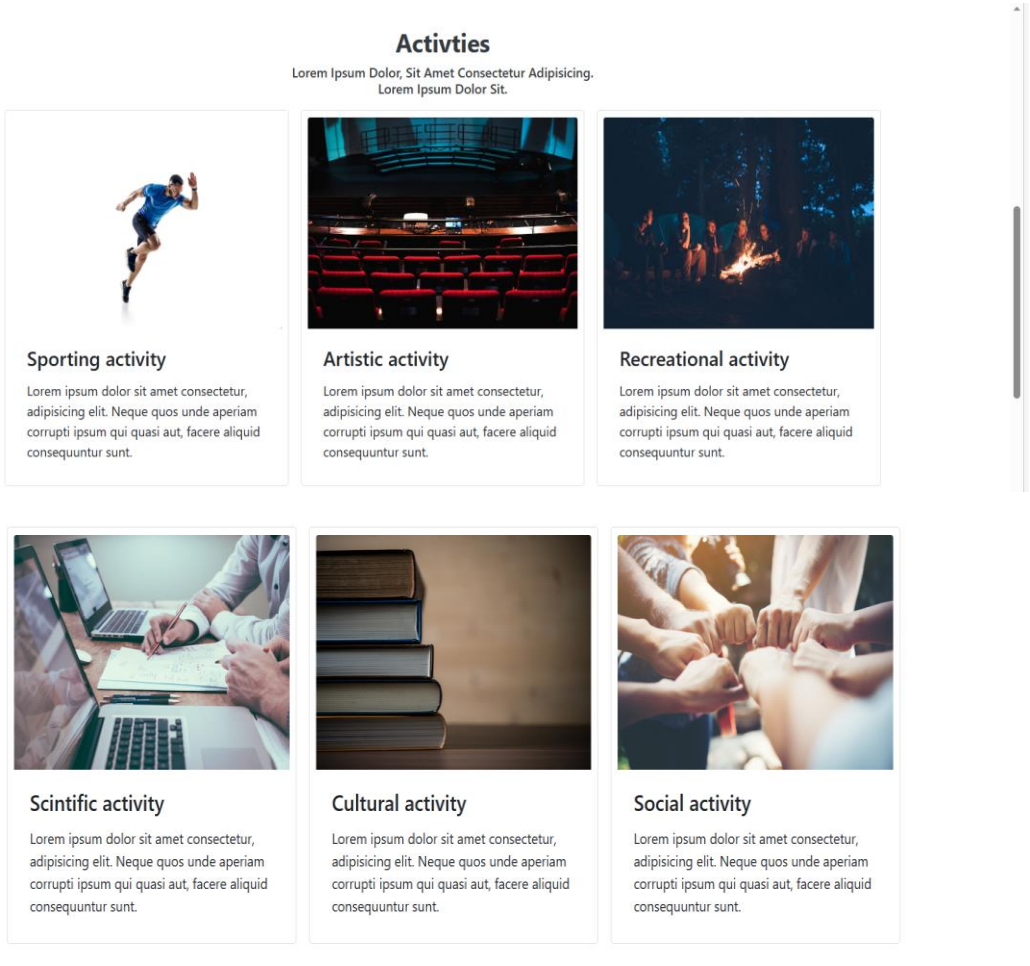


- bar




- Header And button to go to About page


## 3.Cards of activities section:



## 4.Break to write more information

**Lorem ipsum dolor sit amet consectetur adipisicing elit. Quos blanditiis animi ea.**
**Lorem ipsum dolor sit amet consectetur.**

## 5.cards about Our services & our team

### Our Services

Lorem ipsum dolor sit, amet consectetur adipisicing elit. Numquam Lorem ipsum dolor sit amet consectetur Lorem ipsum dolor sit amet consectetur adipisicing Lorem ipsum dolor sit amet consectetur. Lorem ipsum dolor sit amet consectetur adipisicing Lorem ipsum dolor sit amet consectetur.

**Lorem, ipsum**
Lorem ipsum dolor sit amet consectetur . Lorem ipsum dolor sit.

**Lorem, ipsum**
Lorem ipsum dolor sit amet consectetur . Lorem ipsum dolor sit.

### Our Services

Lorem ipsum dolor sit, amet consectetur adipisicing elit. Numquam Lorem ipsum dolor sit amet consectetur Lorem ipsum dolor sit amet consectetur adipisicing Lorem ipsum dolor sit amet consectetur. Lorem ipsum dolor sit amet consectetur adipisicing Lorem ipsum dolor sit amet consectetur.

**Lorem, ipsum**
Lorem ipsum dolor sit amet consectetur . Lorem ipsum dolor sit.

**Lorem, ipsum**
Lorem ipsum dolor sit amet consectetur . Lorem ipsum dolor sit.

### Our Team

Lorem ipsum, dolor sit amet consectetur adipisicing elit.
Autem tenetur eveniet vel voluptatem a, Lorem ipsum dolor sit amet, consectetur adipisicing elit.
consectetur adipisicing elit

Workers Team

- Button to navigate to Section members

Workers Team

## 6.Social links for the project and link for Fci.Zu

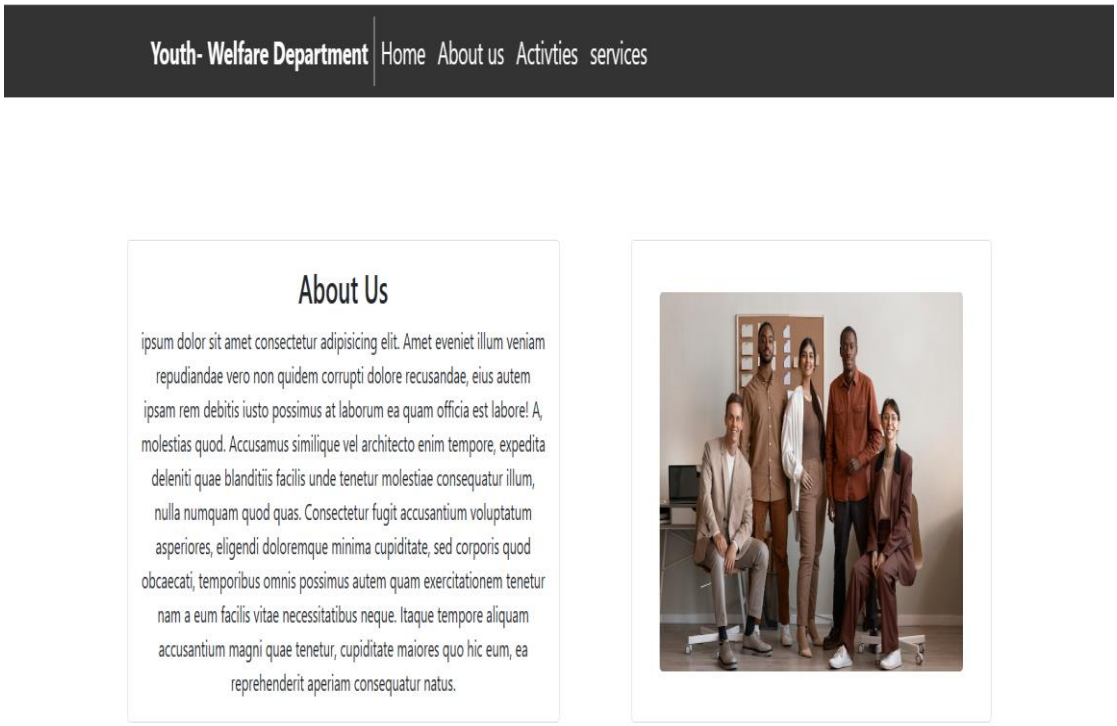
### Contact-Us


Lorem ipsum dolor sit amet consectetur adipisicing elit.
Tempora, ipsum labore. Lorem ipsum dolor, sit amet consectetur adipisicing.
www.Welfare-Departmen.com

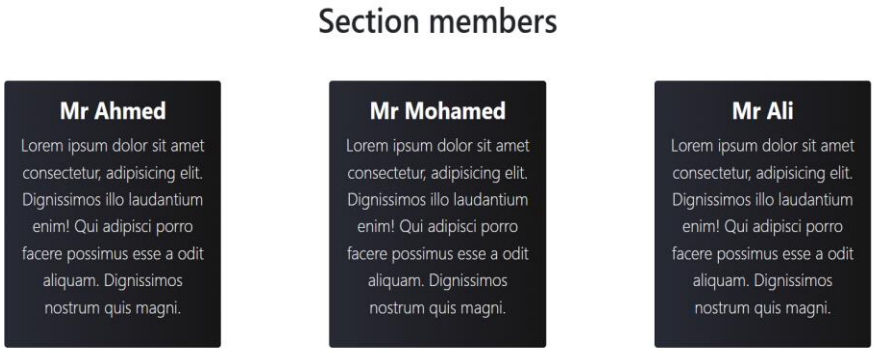
Facebook Linked-in Twitter

## 7. Footer for the project divided to columns

## 8. About us page :

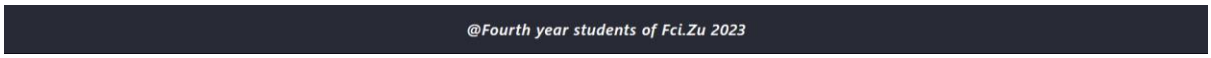



## 9. cards for section members

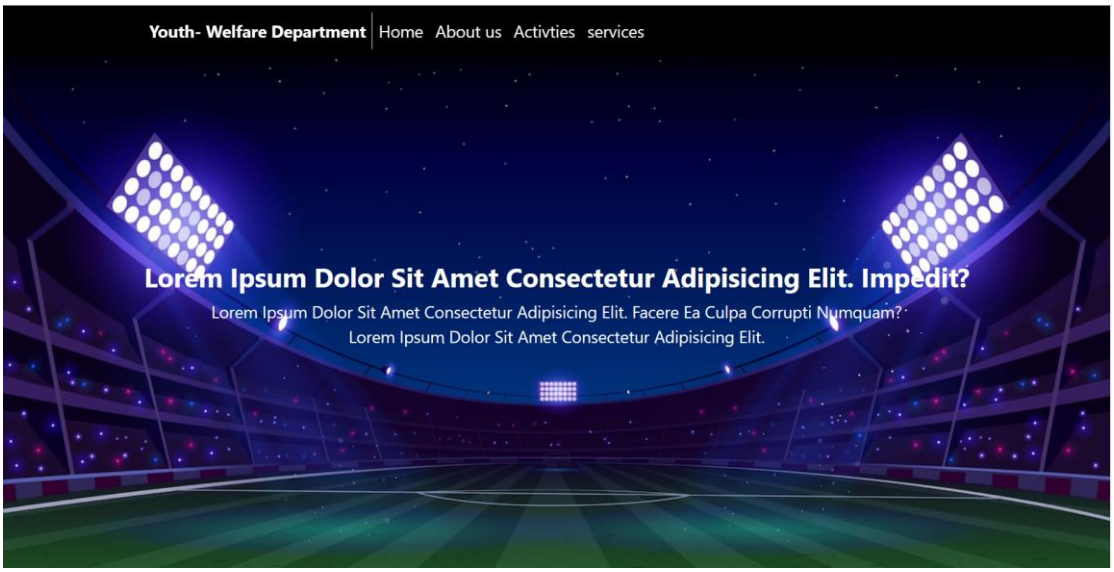


## 10. more information about Our Services


## 11. footer for about page

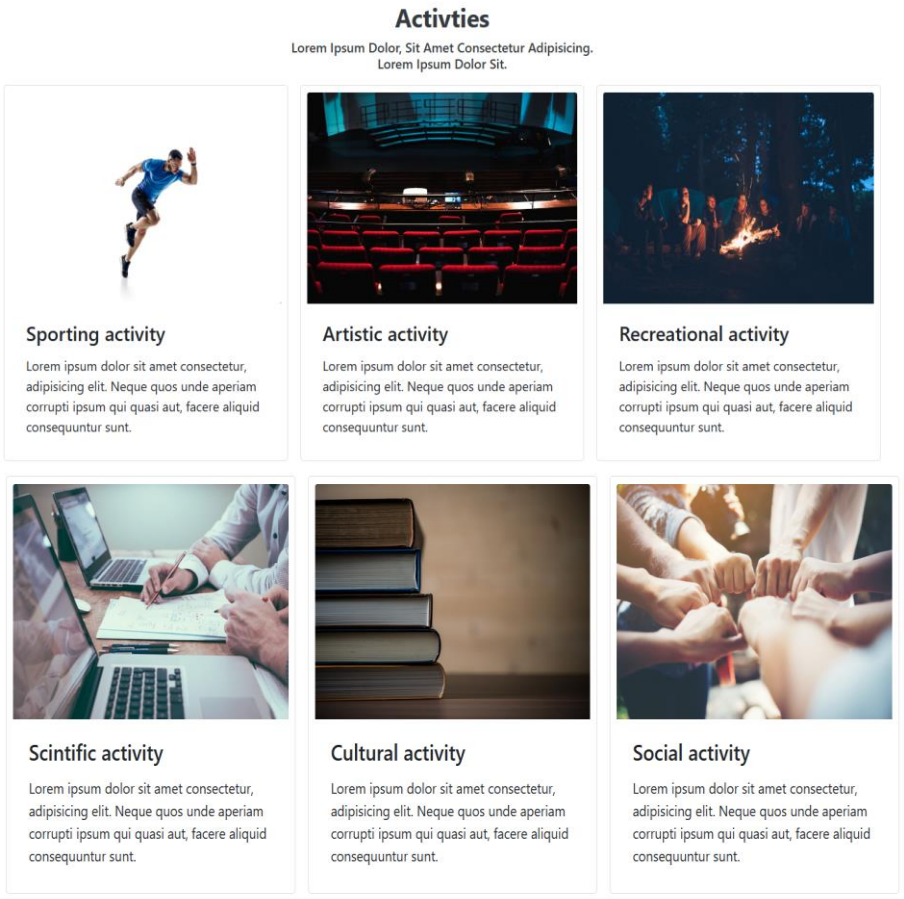



## 12. Activities page

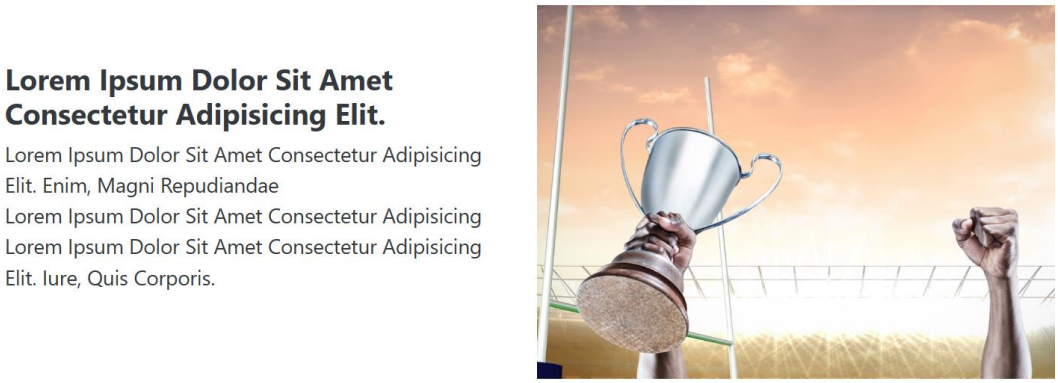


# 13.navbar for activities page


# 14.Cards of activities section:

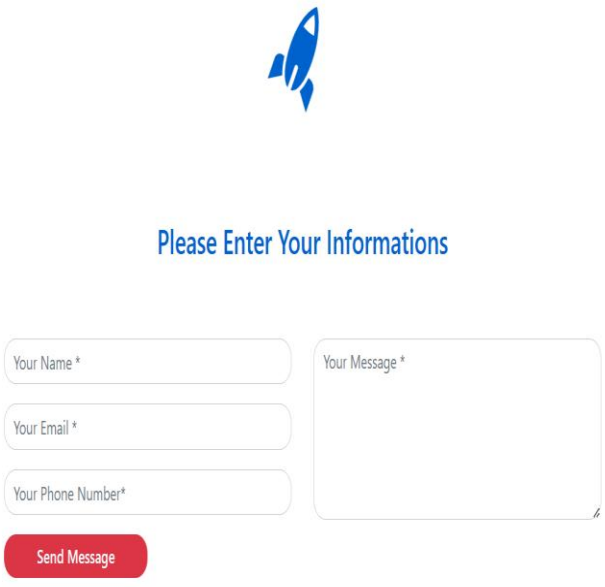



# 15.more information about Activities :



## 16.Services page:


## 17.form for Services page:



# Chapter four

## Project Implementation

# 4.1- Overview:

In his chapter the reader will get implementation of previous with lots of details. This Chapter discusses the system implementation and conversion process and theactivities in this phase.

## System implementation:

System implementation is the process of installing hardware and software andgetting the AIS up and down.

An implementation consists of implementation tasks, expected completion dates, cost estimates and the person or persons responsible for each task. It is during thisphase that the project becomes visible to outsiders, to whom it may appear that theproject has just begun. The implementation phase is the doing phase. The start of the implementation phase involves the creation of database tables before other objects, such as forms and reports, can be added.

## Develop Software Programs:

Seven Steps are followed when developing and testing software programs.

1. Determine user needs.
2. Develop a plan.
3. Write program instructions (code).
4. Test the program.
5. Document the program.
6. Train program users.
7. Install and use the system.

Successful system implementation requires good leadership and careful planning, agood understanding of every component of the system is critical in putting togetheran implementation strategy, Enterprise IT environments involve integration of a variety of vendor technologies. Interoperability standards within commercial software environments are voluntary, and even the simplest system upgrade must be validated at each step of the integration process.

# 4.2- Tools

## Html (Hyper Text Markup Language)

First developed by Tim Berners-Lee in 1990. HTML is short for HypertextMarkup Language. HTML is used to create electronic documents

(called pages) that are displayed on the World Wide Web. Each page contains a series of connections to other pages called hyperlinks. Every web page you see onthe Internet is written using one version of HTML code or another.

HTML code ensures the proper formatting of text and images so that your Internetbrowser may display them as they are intended to look. Without HTML, a browserwould not know how to display text as elements or load images or other elements. 70 Virtual Patient HTML also provides a basic structure of the page, upon which Cascading Style Sheets are overlaid to change its appearance. One could think of HTML as the bones (structure) of a web page, and CSS as its skin (appearance).

## Why to create and view HTML?


Because HTML is a markup language it can be created and viewed in any text editor as long as it is saved with a html file extension. However, most find it easier to design and create web pages in HTML using an HTML editor. Once the HTML file is created it can be viewed locally uploaded to aweb server to be viewed online using a browser.

# CSS

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language. Although most often used to set the visual style of web pages and user interfaces written in HTMLand XHTML, the language can be applied to any XML document, including plain XML, SVG and XUL, and is applicable to rendering in speech, or on other media. Along with HTML and JavaScript, CSS is a cornerstone technology used by mostwebsites to create visually engaging Webpages, user interfaces for web applications, and user interfaces for many mobile applications.

CSS is designed primarily to enable the separation of presentation and content, including aspects such as the layout, colors, and fonts. This separation canimprove content accessibility, provide more flexibility and control in the specification of presentation characteristics, and enable multiple HTML pages to share formatting by specifying the relevant CSS in a separate CSS file, and reduce complexity and repetition in the structural content.

## CSS Syntax

CSS has a simple syntax and uses a number of English keywords to specify the names of various style properties. A style sheet consists of a list of rules. Eachrule or rule-set consists of one or more selectors, and a declaration block.

Selectors: In CSS, selectors declare which part of the markup a style applies to bymatching tags and attributes in the markup itself.


Selectors may apply to:

8. All elements of a specific type.
9. Elements specified by attribute, in particular:
10. Id: an identifier unique within the document
11. Class: an identifier that can annotate multiple elementsin a document.
12. elements depending on how they are placed relative to others

# JavaScript

JavaScript (often shortened to JS) is a lightweight, interpreted, object- oriented language with first-class functions, and is best known as the scripting language for Web pages, but it's used in many non-browser environments as well. It is a prototype-based, multi-paradigm scripting language that is dynamic, and supports object-oriented, imperative, and functional programming styles.

JavaScript runs on the client side of the web, which can be used to design / program how the web pages behave on the occurrence of an event. JavaScript is an easy to learn and also powerful scripting language, widely used for controlling web page behavior.

JavaScript is not "Interpreted Java".

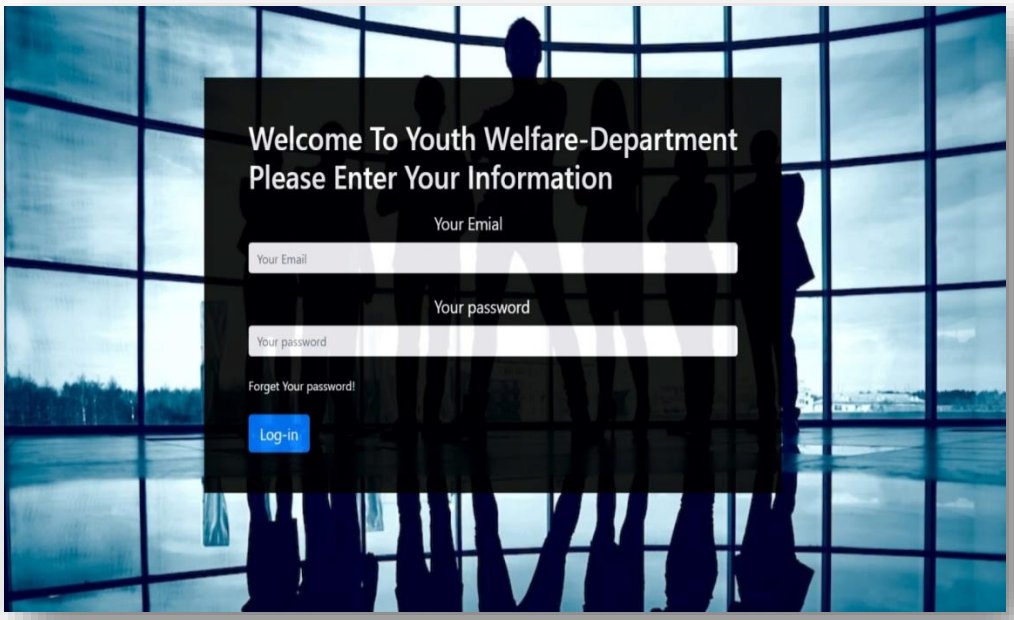
In a nutshell, JavaScript is a dynamic scripting language supporting prototype based object construction. The basic syntax is intentionally similar to both Java and C++ to reduce the number of new concepts required to learn the language.

Language constructs, such as if statements, for and while loops, and switch and try... catch blocks

Function the same as in these languages (or nearly so).

At the first we used design languages (html,css ) to built our project

We make log –in page for students to enters the project



```html
<header>
    <!--start header-->
    <!--start form header login-->
    <div class="form-header">
        <div class="container">
          <div class="row">
           <div class="col-12">
         <h1>
                Welcome To Youth Welfare-Department
          <br>
          Please Enter Your Information
         </h1>
        <label>
           Your Emial
        </label>
        <br>
        <input class="form-control"  type="email" placeholder="Your Email " required>
        <br>
       <label>Your password</label>
        <br>
        <input class="form-control"  type="password" placeholder="Your password" required><
        <a id="forget" href="#">Forget Your password!</a><br><br>
        <button type="button" class="btn btn-primary btn-lg">Log-in</button>

           </div>
          </div>
        </div>
    </div>
</header>
```

## by using CSS language :

select suitable background image for the project and choose suitable color for font and items .

style > # style.css > body

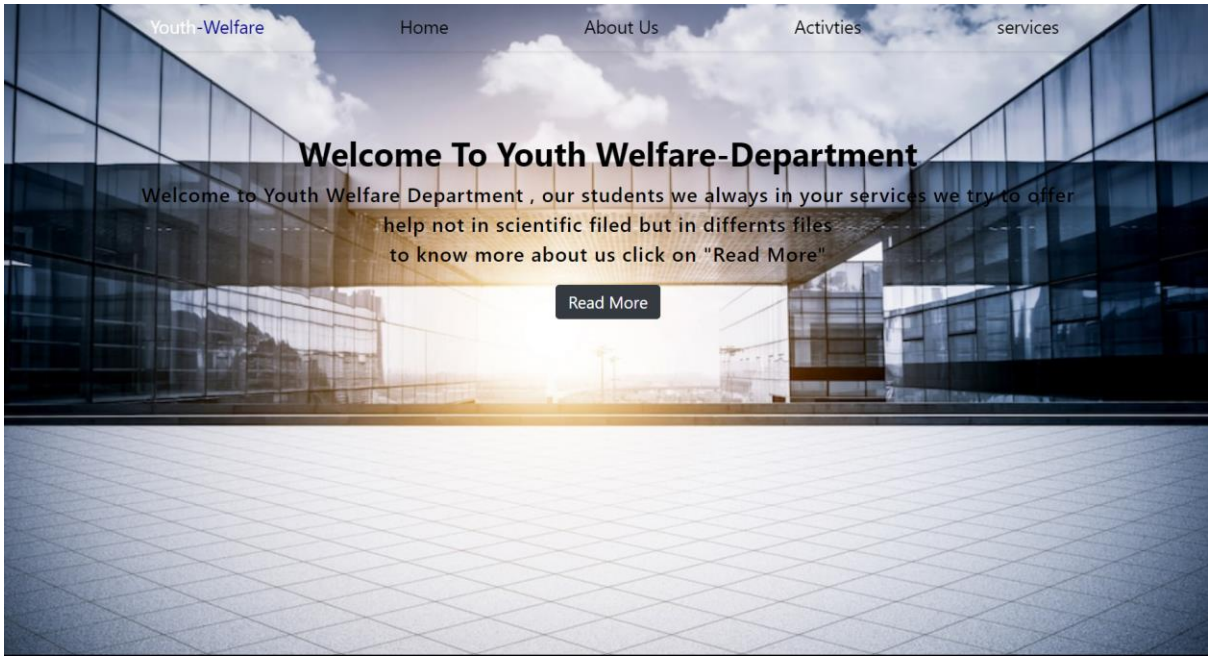
```css
*{
    margin: 0%;
    padding: 0%;
    box-sizing: border-box;
}
body{
    font-family: 'Noto Kufi Arabic', sans-serif;
    font-family: 'Roboto', sans-serif;
}
/*start header*/
header{
    background-image: url(img/header\ background.jpg);
    background-size: cover;
    height: 100vh;
    width: 100%;
}
@media (max-width :575px){
    header{
        background-image: url(img/header\ background.jpg);
        background-size: cover;
        background-position: center;
        height: 100vh;
        width: 100%;
    }
}
.form-header{
    position: absolute;
    margin-top:6%;
    margin-left: 20%;
    background-color: rgb(0, 0, 0);
    opacity: 90%;
    padding: 50px 50px;

}
```

by using media query in' CSS the page responsive
for the users

```css
@media (max-width:575px){
 .form-header{
 width: auto;
 margin-left: 0%;
 }
}
h1{
    color: white;
    font-size: 25px;
    font-weight: bold;
    padding-bottom: 10px;
}

label {
    color: white;
    font-size: 22px;
  margin-left: 38%;
}
#forget{
    color: white;
    font-size: 16px;

}
/*end header*/
```

We make home page describe our project :

In this page we had compelete description for the project



Html code for header section : in header section we have navbar that contains all sections in the project

```
<header>
    <!--start navbar-->
    <nav class="navbar navbar-expand-lg navbar-light shadow-sm navbar-dark fixed-top sticky-top">
     <div class="container Logo">

      <a class="navbar-brand" href="index.html">Youth<span>-Welfare</span></a>

      <a href="index.html">Home </a>
      <a href="about.html"  target="_blank" > About Us </a>
      <a href="Activties.html"  target="_blank" > Activties </a>

      <a  href="services.html" target="_blank">services</a>
      <!--end navbar-->
      </nav>
      <!--strt header text-->
      <div class="col-12" id="header-text">
            <h1 class="h-text">
              Welcome To Youth Welfare-<span>Department</span>
            </h1>
            <p class="p-text">
              Lorem ipsum dolor sit amet consectetur,adipisicing elit. Id, tempora

                  Lorem ipsum dolor, sit amet consectetur adipisicing<br>
            lorem ipsum dolor sit amet
            </p>
            <a id="btn-head"  class="btn btn-dark btn-lg" target="_blank" href="about.html">Read More</a>
          </div>
        </div>
      </div>
     </div>
   <!--end header text-->
</header>
```

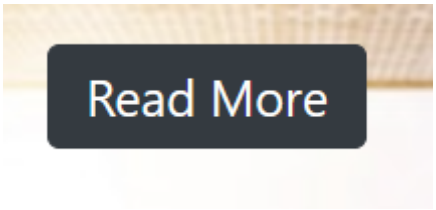
## css code for header section :

by using bootstrap libirary and css code we Coordinating the colors and fonts

```css
header {
    background-image: url(img/empty-floor.jpg);
    background-size: cover;
    background-position: center;
    width: 100%;
    height: 100vh;
}
a {
    text-decoration: none;
}

.container a {
    text-decoration: none;
    color: #000;
    font-size: 18px;
    margin-right: 8px;
}

.container a:hover {
    text-decoration: none;
    color: #000;
}

.Logo a {
    font-size: 22px;
}

span {
    color: #00008b;
    font-size: 22px;

}
```

```css
#lg5 {

    padding: 5px 15px;
    font-size: 1.25rem;
    line-height: 1.5;
    border-radius: 0.3rem;
    color: #fff;
}
#header-text {
    position: relative;
    text-align: center;
    top: 100px;
}
h1.h-text {
    color: #333;
    font-size: 40px;
    font-weight: 700;
}
h1.h-text span {
    color: white;
    font-size: 40px;
    font-weight: 700;
}
p.p-text {
    color: white;
    font-size: 24px;
    margin-left: 16px;
    letter-spacing: 1px;
    font-weight: 300;

}
#header-text #btn-head {
    position: absolute;
    left: 41%;
    margin-top: 3px;
    padding: 5px 15px;
}
```

this navabr that contains all items (sections) that users can use to transfers throught.


In home page By clicking on it, we can navigate to a About page

## Our Services

The Youth Welfare Department is not only responsible for the educational
and recreational guidance of students, but is also responsible for many social and individual activities
and encouraging students to take an interest in their studies and their lives.

**Welfare-Departmen missions**

It will receive new students and introduce them to the role of the college from the various scientific and active aspects.

**Welfare-Departmen missions**

Discovering artistic, sports and cultural talents among students and Helping students who are socially and financially unable.

## Our Services

University life is full and rich in science and various activities, and the college is not only a place to receive knowledge, but it is a spacious field for practicing many hobbies
and activities that develop the student's personality
and refine his abilities and talents so that the student is able to face the challenges of practical life after graduation
These activities include serious and meaningful participation in the following:

**Students' Union missions**

Holding many political, literary and cultural seminars to raise the intellectual and cultural level of students.

**Students' Union missions**

Participate in the Student Union so that the student is more expressive of the problems and opinions of his colleagues.

**Youth- Welfare Department** | Home About us Activties services

## About Us

The department is responsible for providing a variety of activities and services to students. These activities and services are designed to enhance the physical, social and intellectual development of students.

Some of the activities and services offered by the Department of Youth Welfare include:
Sports and recreation programs
Cultural and artistic programs
Social and community service programs

# Section members

### Dr Mohamed Abdulaziz

------ Department Manager ------
He is the one who manages the department, supervises the productivity and daily operations that take place within the department, and solves the problems facing the department.

### Mr Reda Hassan

---------- Social Worker ----------
He is the one who provides support and guidance to young people who are facing challenges in their lives and works with a range of people to ensure they have opportunities to reach their full potential.

### Mr Sameh Samir

---------- Sports Specialist ----------
He is responsible for developing and implementing youth sports programs. He is responsible for promoting physical activity, healthy lifestyles, and positive social interactions among young people.

## Welfare Department Services

Counseling and support services: The department offers counseling and support services to students who are facing challenges, such as academic problems, personal problems, or mental health issues.
Sports and Recreational Programs: The department offers sports and recreational programs for students who wish to develop their sports and recreational skills.
Cultural and artistic programs: The department offers cultural and artistic programs to students who want to explore their interests in the arts. These programs include art exhibits, theater productions, and music performances.

# Navbar code for About Page:-

```
<!--start navbar-->
<div class="container-fluid container-fluid1 ">
    <div class="container ">
    <nav class="navbar navbar-expand-lg  navbar-dark fixed-top sticky-top" >
      <span class="navbar-text pr-2 text-white ">Youth- Welfare Department</span>
      <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#collapsibleNavbar">
        <span class="navbar-toggler-icon"></span>
      </button>
      <!-- Links -->
      <div class="collapse navbar-collapse " id="collapsibleNavbar">
      <ul class="navbar-nav me-auto mb-3 mb-lg-0 ">
        <li class="nav-item">
          <a class="nav-link  text-white" href="index.html">Home</a>
        </li>
        <li class="nav-item">
          <a class="nav-link  text-white" href="about.html" target="_self">About us</a>
        </li>
        <li class="nav-item">
          <a class="nav-link  text-white" href="Activties.html">Activties</a>
        </li>
        <li class="nav-item">
          <a class="nav-link  text-white" href="services.html" >services</a>
        </li>
      </ul>
      </div> <!--</navbar-collapse> -->
      </nav>  <!--</nav> -->
      </div><!--</container> -->
    </div>  <!--</container-fluid> -->
```

# Html Code for About Page

```
<section>
<div class="container">
  <div class="row  row1 justify-content-between">
    <div class="card col-md-6 mb-3" style="text-align: center;">
        <p><h3>About Us</h3> ipsum dolor sit amet consectetur adipisicing elit. Amet eveniet illum veniam repudiandae vero non quidem
    </div><!--card-->
    <div class="card col-md-5 mb-3">
      <img class="rounded" src="style/img/people-work.jpg" alt="" width="400px" style="margin: auto;">
    </div> <!--card-->
  </div><!--row-->
</div> <!--container-->

<div class="container mt-5">
  <h1 style="text-align: center;">Section members</h1>
  <div class="row justify-content-between mt-5">
    <div class="card card1 col-md-3 p-3 mb-3">
      <h3 class="text-white">Mr Ahmed</h3>
      <p>Lorem ipsum dolor sit amet consectetur, adipisicing elit. Dignissimos illo laudantium enim! Qui adipisci porro facere possimu
    </div>
    <div class="card card1 col-md-3 p-3 mb-3">
      <h3 class="text-white">Mr Mohamed</h3>
      <p >Lorem ipsum dolor sit amet consectetur, adipisicing elit. Dignissimos illo laudantium enim! Qui adipisci porro facere possin
    </div>
    <div class="card card1 col-md-3 p-3 mb-3">
      <h3 class="text-white">Mr Ali</h3>
      <p >Lorem ipsum dolor sit amet consectetur, adipisicing elit. Dignissimos illo laudantium enim! Qui adipisci porro facere possin
    </div>
  </div>
</div>
```

```
<div class="container">
  <div class="row  row1 justify-content-between">
    <div class="card col-md-6 mb-3" style="text-align: center;">
        <p><h3>Our Services</h3> ipsum dolor sit amet consectetur adipisicing elit. Amet eveniet illum veniam repudiandae vero non qui
    </div><!--card-->
    <div class="card col-md-5 mb-3">
      <img class="rounded" src="style/img/header background.jpg" alt="" width="400px" style="margin: auto;">
    </div> <!--card-->
  </div><!--row-->
</div> <!--container-->
</section>
```

## Css code for about page

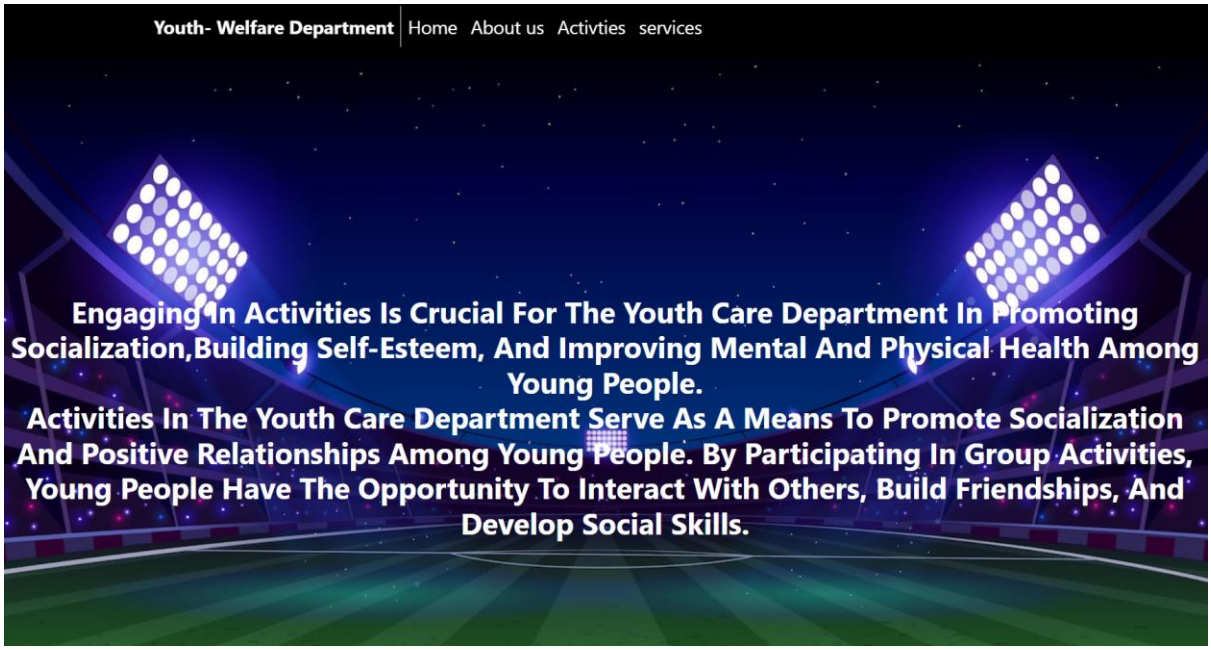
```css
a
{
    text-decoration: none;
}
.d-flex
{
    margin-left: auto;
}
.container-fluid1
{
    background-color: rgba(0, 0, 0, 0.8);
}
.navbar-text
{
border-right: 1px solid white;
font-size: 22px;
font-weight: bold;
 transition: .5s;
 cursor: pointer;
}
.navbar-text:hover
{
    text-shadow: 2px -2px 14px rgb(239, 233, 228);
}
.nav-link
{
font-size: 22px;
}
```

```css
.row1
{
    margin-top: 100px;
}
.card1
{
    background: -webkit-linear-gradient(left, #282A35, rgb(23, 22, 22));
    text-align: center;
}
.card1 h3{
    font-size:28px;
    font-weight: bolder;
}
.card1 p{
    font-size: 21px;
    color: #fff;
    font-weight:300;

}
.card1:hover{
    cursor: pointer;
    box-shadow: 0 8px 8px rgba(0, 0, 0, .26);
    -webkit-transition-duration: 0.3s;
    -moz-transition-duration: 0.3s;
    -ms-transition-duration: 0.3s;
    -o-transition-duration: 0.3s;
    transition-duration: 0.3s;
}
```

in this page we used 'bootstrap library' to coordination by (grid – system)

We divide the element to two (div).

# Activities page




# navbar for activities page

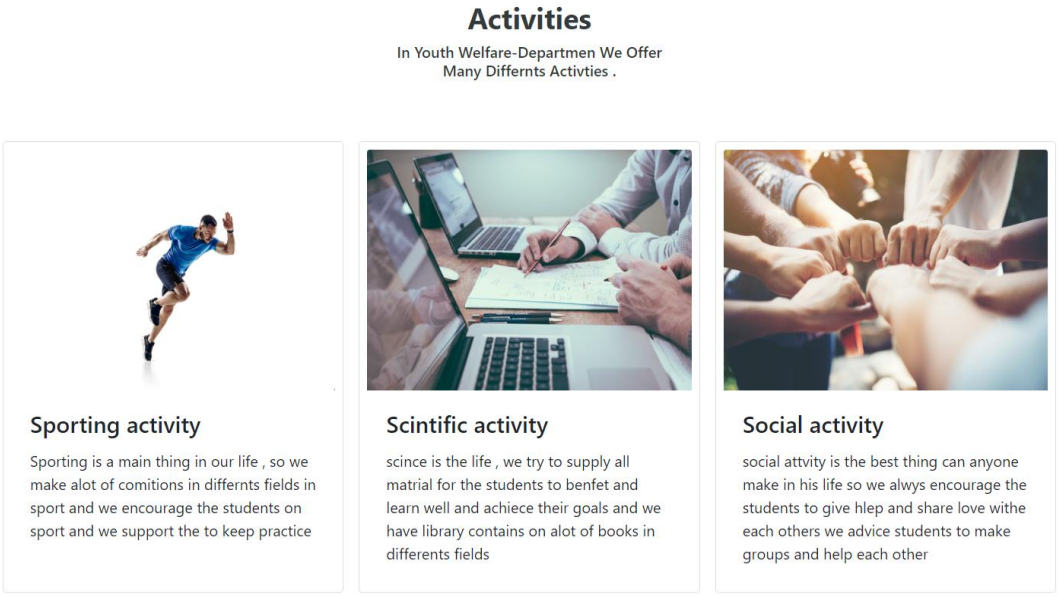

# Html code for navbar:

```
<!--start navbar-->
<header>
<div class="container-fluid container-fluid1 ">
    <div class="container ">
    <nav class="navbar navbar-expand-lg  fixed-top sticky-top" >
      <span class="navbar-text pr-2 text-white ">Youth- Welfare Department</span>
      <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#collapsibleNavbar">
        <span class="navbar-toggler-icon"></span>
      </button>
      <!-- Links -->
      <div class="collapse navbar-collapse " id="collapsibleNavbar">
      <ul class="navbar-nav me-auto mb-3 mb-lg-0 ">
        <li class="nav-item">
          <a class="nav-link  text-white" href="index.html">Home</a>
        </li>
        <li class="nav-item">
          <a class="nav-link  text-white" href="about.html">About us</a>
        </li>
        <li class="nav-item">
          <a class="nav-link  text-white" href="Activties.html" target="_self">Activties</a>
        </li>
        <li class="nav-item">
          <a class="nav-link  text-white" href="services.html" >services</a>
        </li>
      </ul>
      </div> <!--</navbar-collapse> -->
      </nav>  <!--</nav> -->
      </div><!--</container> -->
    </div>
</header>
```

# Css code for navbar:

```css
a
{
    text-decoration: none;
}
.d-flex
{
    margin-left: auto;
}
.container-fluid1
{
    background-color: black;
    border: solid 1px black;
}
.navbar-text
{
border-right: 1px solid white;
font-size: 22px;
font-weight: bold;
 transition: .5s;
 cursor: pointer;
}
.navbar-text:hover
{
    text-shadow: 2px -2px 14px rgb(239, 233, 228);
}
.nav-link
{
font-size: 22px;
}
```

# Cards of activities section:



The Youth Care Department Is An Important Department That Contributes To Creating A Healthy And Positive Learning Environment For Students. This Department Has A Significant Impact And Importance On The College And Students For Several Reasons.

Overall, The Youth Care Department Plays A Vital Role In Enhancing The Academic Experience For Students In The College Of Computer And Information Science, By Supporting Their Well-Being, Fostering A Sense Of Community, And Promoting Personal Growth.

## Html code for section cards:

```
    <div class="container-fluid">
      <div class="row justify-content-center">
          <div class="card m-2  col-lg-3 p-2">
            <img class="card-img-top" src="style/img/scince (1).jpeg" alt="" width="300px" height="250px">
            <div class="card-body">
              <h4 class="card-title">Scintific activity</h4>
              <p class="card-text" >Lorem ipsum dolor sit amet consectetur, adipisicing elit. Neque quos unde aperiam corrupti ipsum c
            </div>
          </div>
          <div class="card m-2  col-lg-3 p-2">
            <img class="card-img-top" src="style/img/Cultural activity.jpg" alt="" width="300px" height="250px">
            <div class="card-body">
              <h4 class="card-title">Cultural activity</h4>
              <p class="card-text">Lorem ipsum dolor sit amet consectetur, adipisicing elit. Neque quos unde aperiam corrupti ipsum qu

            </div>
          </div>
          <div class="card m-2  col-lg-3 p-2">
            <img class="card-img-top" src="style/img/social.jpg" alt="" width="300px" height="250px">
            <div class="card-body">
              <h4 class="card-title">Social activity</h4>
              <p class="card-text">Lorem ipsum dolor sit amet consectetur, adipisicing elit. Neque quos unde aperiam corrupti ipsum qu

            </div>
          </div>
      </div>
  </div><!--end card2-->
```

```
<!--start card sections-->
<div class="col-md-12 activties">
  <h2 class="text-dark text-capitalize">Activties</h2>
  <h6 class="text-dark text-capitalize"> Lorem ipsum dolor, sit amet consectetur adipisicing. <br>
    Lorem ipsum dolor sit.</h6>
</div>
<div class="container-fluid">
  <div class="row justify-content-center">
      <div class="card m-2  col-lg-3 p-2">
        <img class="card-img-top" src="style/img/sport.jpg" alt="" width="300px" height="250px">
        <div class="card-body">
          <h4 class="card-title">Sporting activity</h4>
          <p class="card-text" >Lorem ipsum dolor sit amet consectetur, adipisicing elit. Neque quos unde aperiam corrupti ipsum c
        </div>
      </div>
      <div class="card m-2  col-lg-3 p-2">
        <img class="card-img-top" src="style/img/artistic activity.jpg" alt="" width="300px" height="250px">
        <div class="card-body">
          <h4 class="card-title"> Artistic activity</h4>
          <p class="card-text">Lorem ipsum dolor sit amet consectetur, adipisicing elit. Neque quos unde aperiam corrupti ipsum qu

        </div>
      </div>
      <div class="card m-2  col-lg-3 p-2">
        <img class="card-img-top" src="style/img/Recreational activity1.jpeg" alt="" width="300px" height="250px">
        <div class="card-body">
          <h4 class="card-title">Recreational activity</h4>
          <p class="card-text">Lorem ipsum dolor sit amet consectetur, adipisicing elit. Neque quos unde aperiam corrupti ipsum qu

        </div>
      </div>
</div>
```

```
<div class="container champions">
  <div class="row">
    <div class="col-md-6 text">
      <h4 class="text-dark text-capitalize">Lorem ipsum dolor sit amet consectetur adipisicing elit.</h4>
      <p class="text-dark text-capitalize">Lorem ipsum dolor sit amet consectetur adipisicing elit. Enim, magni repudiandae <br>
      Lorem ipsum dolor sit amet consectetur adipisicing <br>
    Lorem ipsum dolor sit amet consectetur adipisicing elit. Iure, quis corporis.</p>
    </div>
    <div class="col-md-6 ">
      <img src="style/img/sportsman_1134-1020.jpg" class="mb-5" width="570px" height="400px">
    </div>
  </div>
</div>
```

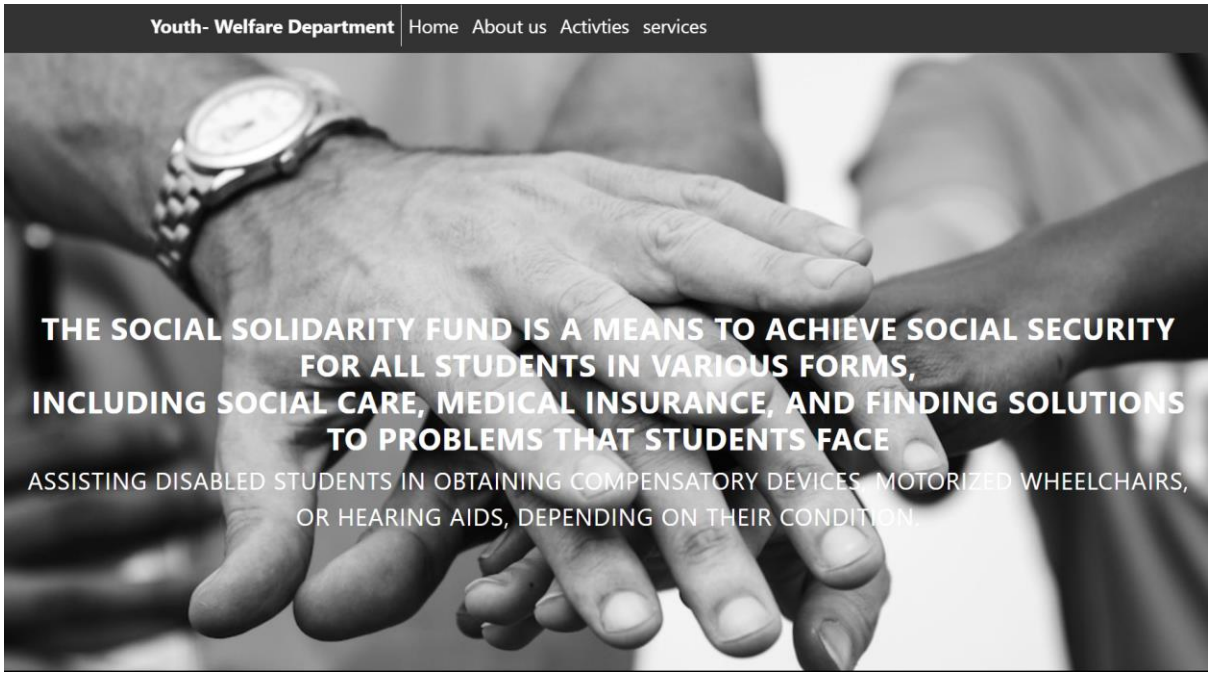
# CSS code:

```css
section{
    background-image: url(img/realistic-soccer.jpg);
    background-position: center;
    background-size: cover;
    height: 90vh;
    width: 100%;

}
section .col-12{
    position: relative;
    text-align: center;
    top: 40%;
}
.col-12 h2{
    font-size: 35px;
    font-weight: bolder;
}
.col-12 p{
    font-size: 22px;
    font-weight: 400;
}
.col-md-12.activties{
    position: relative;
    text-align: center;
    margin-top: 30px;

}
.col-md-12.activties h2{
    font-size: 30px;
    font-weight: bolder;
}
.col-md-12.activties p{
    font-size: 25px;
    font-weight: bolder;
}
```

```css
.card:hover{
    cursor: pointer;
    box-shadow: 0 8px 8px rgba(0, 0, 0, .26);
    -webkit-transition-duration: 0.3s;
    -moz-transition-duration: 0.3s;
    -ms-transition-duration: 0.3s;
    -o-transition-duration: 0.3s;
    transition-duration: 0.3s;
}
/*end card effects*/
.container.champions{
    margin-top: 40px;
}
.col-md-6.text{
    margin-top: 60px;
}
.col-md-6.text h4{
    font-size: 31px;
    font-weight: bolder;
    color: black;

}
.col-md-6.text p{
    font-size: 22px;
    font-weight: 400
}
@media (max-width:575px){
    .col-md-6 img{
        width:400px ;
        height: 350px;
    }
}
```

We used media query and bootstrap to responsive this page

Services page:




Please Enter Your Informations

Your Name *

Your Email *

Your Phone Number*

Your Message *

Send Message

# Html code:

```
<div class="container-fluid container-fluid1 ">
    <div class="container ">
    <nav class="navbar navbar-expand-lg  navbar-dark fixed-top sticky-top" >
      <span class="navbar-text pr-2 text-white ">Youth- Welfare Department</span>
      <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#collapsibleNavbar">
        <span class="navbar-toggler-icon"></span>
      </button>
      <!-- Links -->
      <div class="collapse navbar-collapse " id="collapsibleNavbar">
      <ul class="navbar-nav me-auto mb-3 mb-lg-0 ">
        <li class="nav-item">
          <a class="nav-link  text-white" href="index.html">Home</a>
        </li>
        <li class="nav-item">
          <a class="nav-link  text-white" href="about.html" target="_self">About us</a>
        </li>
        <li class="nav-item">
          <a class="nav-link  text-white" href="Activties.html">Activties</a>
        </li>
        <li class="nav-item">
          <a class="nav-link  text-white" href="#" >services</a>
        </li>
      </ul>
      </div> <!--</navbar-collapse> -->
      </nav>  <!--</nav> -->
      </div><!--</container> -->
    </div>  <!--</container-fluid> -->
```

```
<div class="container-fluid header">
  <div class="col-md-12 text">
    <h2 class="text-white font-weight-bolder ">
      Lorem ipsum dolor sit amet consectetur adipisicing<br>
      Lorem ipsum dolor sit amet consectetur.
    </h2>
    <p class="text-white">
      Lorem ipsum dolor sit amet consectetur adipisicing elit. Cum, voluptas. Minima, obcaecati similique.
    </p>
  </div>
</div>
<section>
  <div class="container contact-form">
    <div class="contact-image">
      <img src="style/img/rocket_contact.png" alt="rocket_contact"/>
    </div>
    <form method="post">
      <h3>Please Enter Your Informations</h3>
      <div class="row">
        <div class="col-md-6">
          <div class="form-group">
            <input type="text" name="txtName" class="form-control" placeholder="Your Name *" value="" />
          </div>
          <div class="form-group">
            <input type="text" name="txtEmail" class="form-control" placeholder="Your Email *" value="" />
          </div>
          <div class="form-group">
            <input type="tel" maxlength="20" name="password" class="form-control" placeholder="Your Phone Number*" value="" />
          </div>
          <div class="form-group">
            <input type="submit" name="btnSubmit" class="btnContact" value="Send Message" />
          </div>
        </div>
        <div class="col-md-6">
          <div class="form-group">
            <textarea name="txtMsg" class="form-control" placeholder="Your Message *" style="width: 100%; height: 150px;"></te
          </div>
```

## CSS code :

```css
a
{
    text-decoration: none;
}
.d-flex
{
    margin-left: auto;
}
.container-fluid1
{
    background-color: rgba(0, 0, 0, 0.8);
}
.navbar-text
{
border-right: 1px solid white;
font-size: 22px;
font-weight: bold;
 transition: .5s;
 cursor: pointer;
}
.navbar-text:hover
{
    text-shadow: 2px -2px 14px rgb(239, 233, 228);
}
.nav-link
{
font-size: 22px;
}
/* end header and navbar*/
.container-fluid.header{
    position: relative;
    text-align: center;
background-image: url(img/team-business.jpg);
background-position: center;
background-size: cover;
height: 100vh;
width: 100%;
}
```

```css
.container-fluid.header{
    position: relative;
    text-align: center;
background-image: url(img/team-business.jpg);
background-position: center;
background-size: cover;
height: 100vh;
width: 100%;
}
.col-md-12.text{
    position: relative;
    top: 38%;
}
@media (max-width:575px){
    .col-md-12.text{
        position: relative;
        top: 10%;
        font-size: 23px;
    }
}
.col-md-12.text h2{
    font-size: 36px;
    letter-spacing: 1px;
    text-transform: uppercase;
}
.col-md-12.text p{
    font-size: 28px;
    letter-spacing: 1px;
    text-transform: uppercase;
}

section{
    width: 100%;
    height: 78vh;
}
```

```css
.contact-form{
    background: #fff;
    margin-top: 10%;
    margin-bottom: 5%;
    width: 70%;
}
.contact-form .form-control{
    border-radius:1rem;
}
.contact-image{
    text-align: center;
}
.contact-image img{
    border-radius: 6rem;
    width: 11%;
    margin-top: -3%;
    transform: rotate(29deg);
}
.contact-form form{
    padding: 14%;
}
.contact-form form .row{
    margin-bottom: -7%;
}
.contact-form h3{
    margin-bottom: 8%;
    margin-top: -10%;
    text-align: center;
    color: #0062cc;
}
```

```css
.contact-form .btnContact {
    width: 50%;
    border: none;
    border-radius: 1rem;
    padding: 1.5%;
    background: #dc3545;
    font-weight: 600;
    color: #fff;
    cursor: pointer;
}
.btnContactSubmit
{
    width: 50%;
    border-radius: 1rem;
    padding: 1.5%;
    color: #fff;
    background-color: #0062cc;
    border: none;
    cursor: pointer;
}
```

## Students' Union

Student unions work to prepare cadres capable of assuming responsibility,
consolidating national awareness, raising the value of belonging, deepening the foundations of democracy
and human rights, as well as facilitating the practice of students expressing
their opinions freely on various issues.

## Contact-Us


To contact with us this our social page and we alwys in students services

Students' Union
WWW.fci.zu.com

# Social links for the project and link for Fci.Zu



# Footer for the project divided to columns .

# Welfare-department

In the project we used html and css for build and design the project and used bootstrap library for coordinate the content of the project .

```
<!--bootstrap css-->
<link rel="stylesheet" href="style/bootstrap.min.css">
```

```
<!--Bootstrap files-->
<script src="js/jquery-3.5.1.slim.min.js"></script>
<script src="js/popper.min.js"></script>
<script src="js/bootstrap.min.js"></script>
```

And used google fonts library for text

```
<!--google font-->
<link rel="preconnect" href="https://fonts.googleapis.com">
<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
<link href="https://fonts.googleapis.com/css2?family=Noto+Kufi+Arabic:wght@900&family=Roboto:wght@300;400;700&display=swap" rel="style
<link href="https://fonts.googleapis.com/css2?family=Roboto:wght@300;400;700&display=swap" rel="stylesheet">
```

Then we used font-awesom for icons

```
<!--font awesom-->
<link rel="stylesheet" href="font
web/css/all.min.css">
```

```
<!--font awesom-->
<script src="fontawesome-free-5.15.4-web/js/all.min.js"></script>
```

# The Part of Database

## What is a Database?

- A **database** is information that is set up for easy access, management and updating. Computer databases typically store aggregations of data records or files that contain information, such as sales transactions, customer data, financials and product information.

- **Databases** are used for storing, maintaining and accessing any sort of data. They collect information on people, places or things. That information is gathered in one place so that it can be observed and analyzed. Databases can be thought of as an organized collection of information.

- a **database** is an organized collection of data (also known as a data store) stored and accessed electronically through the use of a database management system. Small databases can be stored on a file system, while large databases are hosted on computer clusters or cloud storage. The design of databases spans formal techniques and practical considerations, including data modeling, efficient data representation and storage, query languages, security and privacy of sensitive data, and distributed computing issues, including supporting concurrent access and fault tolerance.
- A **database management system** (**DBMS**) is the software that interacts with end users, applications, and the database itself to capture and analyze the data. The DBMS software additionally encompasses the core facilities provided to administer the database. The sum total of the database, the DBMS and the associated applications can be referred to as a **database system**. Often the term "database" is also used loosely to refer to any of the DBMS, the database system or an application associated with the database.
- Computer scientists may classify database management systems according to the database models that they support. Relationa databases became dominant in the 1980s. These model data as rows and columns in a series of tables, and the vast majority use SQL for writing and querying data. In the 2000s, non-relational databases became popular, collectively referred to as NoSQL, because they use different query languages.

- **What are databases used for?**

Businesses use data stored in databases to make informed business decisions. Some of the ways organizations use databases include the following:

Improve business processes. Companies collect data about business processes, such sales, order processing and customer service. They analyze that data to improve these processes, expand their business and grow revenue.

Keep track of customers. Databases often store information about people, such as customers or users. For example, social media platforms use databases to store user information, such as names, email addresses and user behavior. The data is used to recommend content to users and improve the user experience…..Secure personal health information. Healthcare providers use databases to securely store personal health data to inform and improve patient care Store personal data. Databases can also be used to store personal information. For example, personal cloud storage is available for individual users to store media, such as photos, in a managed cloud.

# Evolution of databases:

Databases were first created in the 1960s. These early databases were network models where each record is related to many primary and secondary records. Hierarchical databases were also among the early models. They have tree schemas with a root directory of records linked to several subdirectories.

Relational databases were developed in the 1970s. Object-oriented databases came next in the 1980s. Today, we use Structured Query Language (SQL), NoSQL and cloud databases.

## • Types of databases :

There are many types of databases. They may be classified according to content type: bibliographic, full text, numeric and images. In computing, databases are often classified based on the organizational approach they use.

Some of the main organizational databases include the following:

**Relational.** This tabular approach defines data so it can be reorganized and accessed in many ways. Relational databases are comprised of tables. Data is placed into predefined categories in those tables. Each table has columns with at least one data category, and rows that have a certain data instance for the categories which are defined in the columns. Information in a relational database about a specific customer is organized into rows, columns and tables. These are indexed to make it easier to search using SQL or NoSQL queries.

Relational databases use SQL in their user and application program interfaces. A new data category can easily be added to a relational database without having to change the existing applications. A relational database management system (RDBMS) is used to store, manage, query and retrieve data in a relational database.

Typically, the RDBMS gives users the ability to control read/write access, specify report generation and analyze use. Some databases offer atomicity, consistency, isolation and durability, or ACID, compliance to guarantee that data is consistent and that transactions are complete.

## Distributed :

This database stores records or files in several physical locations. Data processing is also spread out and replicated across different parts of the network.

Distributed databases can be homogeneous, where all physical locations have the same underlying hardware and run the same operating systems and database applications. They can also be heterogeneous. In those cases, the hardware, OS and database applications can be different in the various locations

## Cloud :

These databases are built in a public, private or hybrid cloud for a virtualized environment. Users are charged based on how much storage and bandwidth they use. They also get scalability on demand and high availability. These databases can work with applications deployed as software as a service.

## Nosql :

NoSQL databases are good when dealing with large collections of distributed data. They can address big data performance issues better than relational databases. They also do well analyzing large unstructured data sets and data on virtual servers in the cloud. These databases can also be called non-relational databases.

## Object_orinted :

These databases hold data created using object-oriented programming languages. They focus on organizing objects rather than actions and data rather than logic. For instance, an image data record would be a data object, rather than an alphanumeric value.

# Graph:

These databases are a type of NoSQL database. They store, map and query relationships using concepts from graph theory. Graph databases are made up of nodes and edges. Nodes are entities and connect the nodes.

These databases are often used to analyze interconnections. Graph databases are often used to analyze data about customers as they interact with a business on webpages and in social media.

Graph databases use SPARQL, a declarative programming language and protocol, for analytics. SPARQL can perform all the analytics that SQL can perform, and can also be used for semantic analysis, or the examination of relationships. This makes it useful for performing analytics on data sets that have both structured and unstructured data. SPARQL lets users perform analytics on information stored in a relational database, as well as friend-of-a-friend relationships, PageRank and shortest path.

## Graph database vs. relational database

| | Graph database | Relational database |
|---|---|---|
| FORMAT | Nodes and edges | Tables with rows and columns |
| RELATIONSHIPS | Considered data, represented by edges between nodes | Related across tables, established using foreign keys between tables |
| COMPLEX QUERIES | Run quickly and do not require joins | Require complex joins between tables |
| TOP USE CASES | Relationship-heavy use cases, including fraud detection and recommendation engines | Transaction-focused use cases, including online transactions and accounting |
| EXAMPLE |  |  |

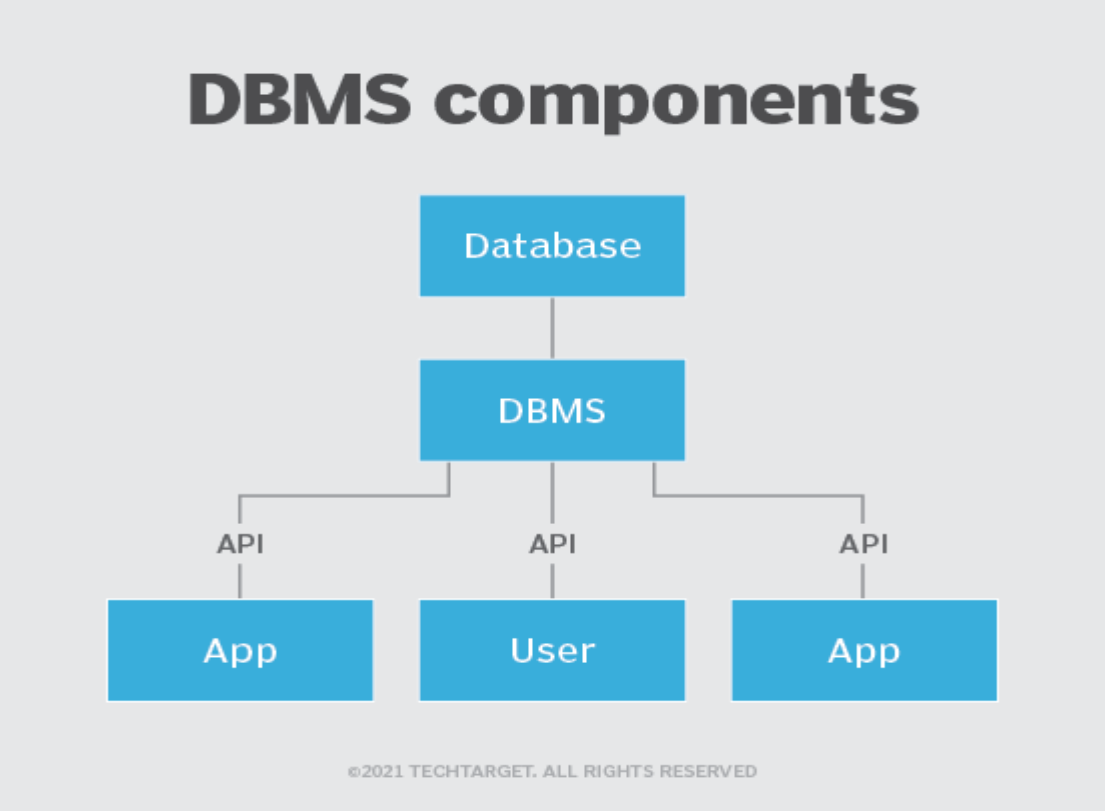


# What are the components of a database?

While the different types of databases vary in schema, data structure and data types most suited to them, they are all comprised of the same five basic components.

1. **Hardware.** This is the physical device that database software runs on. Database hardware includes computers, servers and hard drives.
2. **Software.** Database software or application gives users control of the database. Database management system (DBMS) software is used to manage and control databases.
3. **Data.** This is the raw information that the database stores. Database administrators organize the data to make it more meaningful.

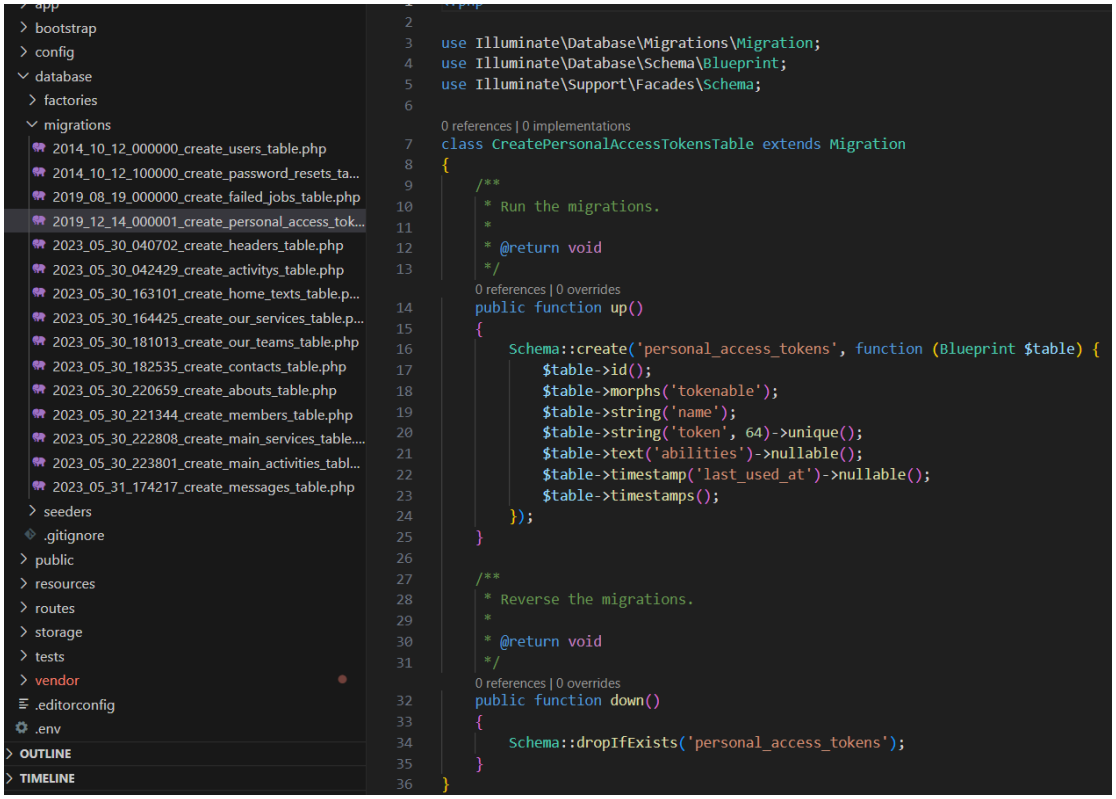
4. **Data access language.** This is the programming language that controls the database. The programming language and the DBMS must work together. One of the most common database languages is SQL.
5. **Procedures.** These rules determine how the database works and how it handles the data.



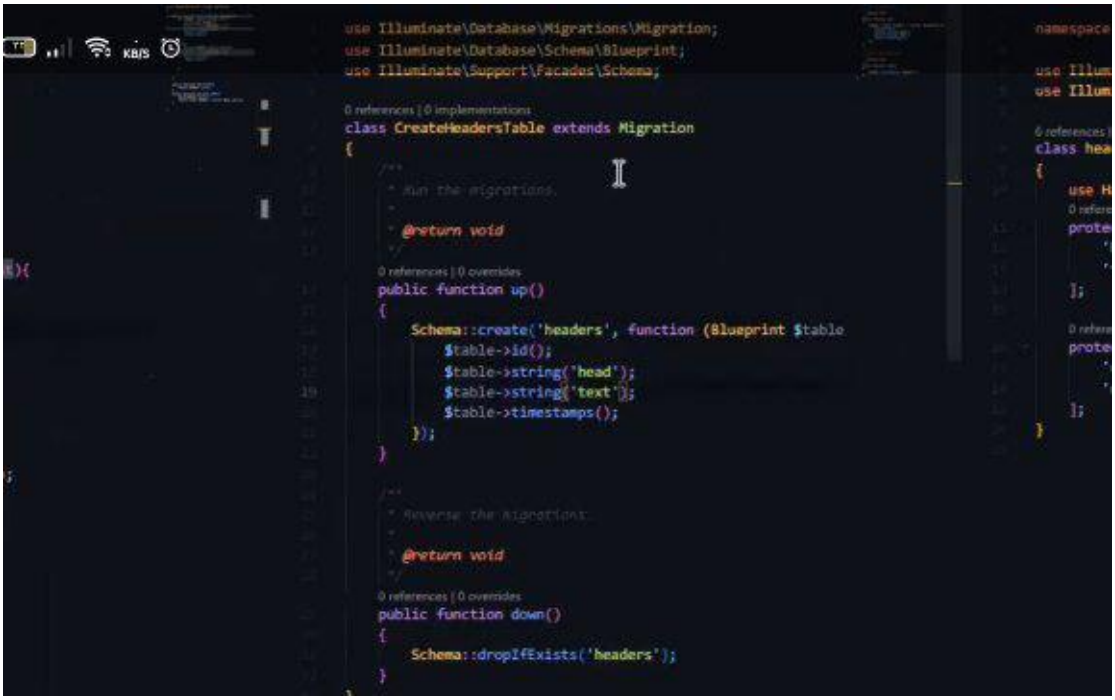

## What is a database management system?

A DBMS enables users to create and manage a database. It also helps users create, read, update and delete data in a database, and it assists with logging and auditing functions.

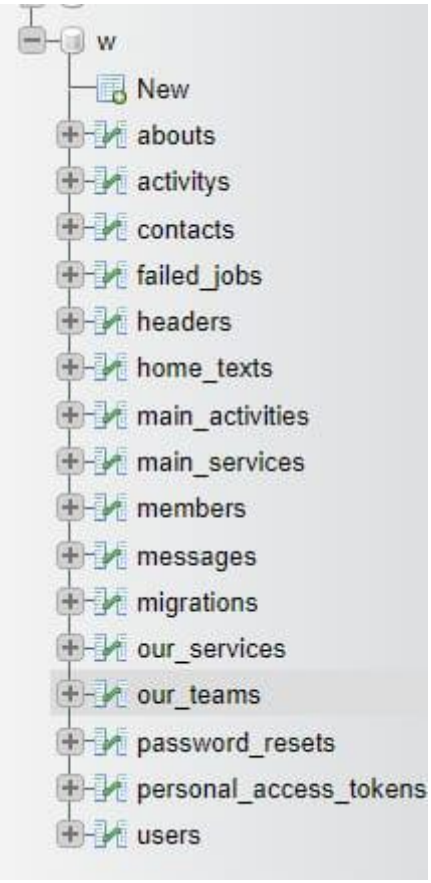
The DBMS provides physical and logical independence from data. Users and applications do not need to know either the physical or logical locations of data. A DBMS can also limit and control access to the database and provide different views of the same database schema to multiple users.




-migrations is the place table which contain any table in database and it is very important for the project.

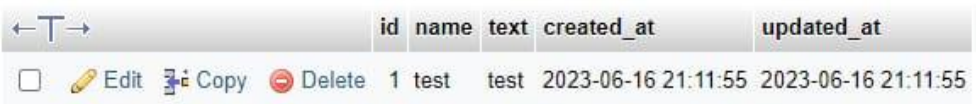


-in migrations we made a table called "headers" then we added two column the frist called 'head' and the oyher called 'text'….after that we will add the two columns to the file named model.




This is all tables we used in data base in our project.

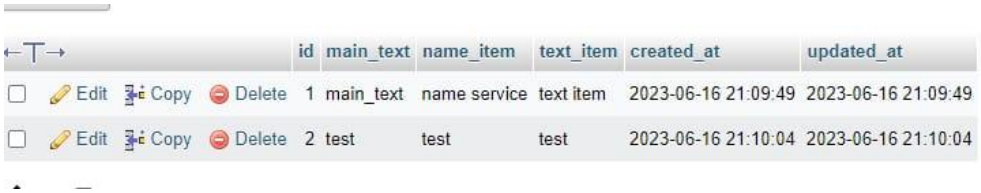

-this table named our_team the first column is about 'id' and the second column is about 'text' and the other column is about 'image'.




-this table named members the first column is about 'id' and the second column is about 'name' and the other column is about 'text'.

| | id | name | email | phone | message | created_at | updated_at |
|---|---|---|---|---|---|---|---|
| Edit Copy Delete | 1 | mahmoud | email@gmail.com | 01220458853 | test | 2023-06-16 21:13:58 | 2023-06-16 21:13:58 |
| Edit Copy Delete | 2 | test | hogay23242@dilanfa.com | +201220458853 | text | 2023-06-16 21:14:48 | 2023-06-16 21:14:48 |

-this table named 'messages' the first column is about 'id' and the second column is about 'name' and the third column is about 'email' and the fourth column is about 'phone' and the other column is about 'message'.



| | id | main_text | name_item | text_item | created_at | updated_at |
|---|---|---|---|---|---|---|
| Edit Copy Delete | 1 | main_text | name service | text item | 2023-06-16 21:09:49 | 2023-06-16 21:09:49 |
| Edit Copy Delete | 2 | test | test | test | 2023-06-16 21:10:04 | 2023-06-16 21:10:04 |

-this table named 'our services' the frist column is about 'id' and the second column is about 'main text' and the third column is about 'name-item' and the fourth column is about 'text_item' .

| | id | text | created_at | updated_at |
|---|---|---|---|---|
| Edit Copy Delete | 1 | odai | 2023-06-16 21:09:06 | 2023-06-16 21:16:58 |

-this table named 'home text' the first column is about 'id' and the second column is about 'text' .

| | id | text | description | image | created_at | updated_at |
|---|---|---|---|---|---|---|
| Edit Copy Delete | 1 | test text | description | 1686949996wallpaperflare.com_wallpaper (1).jpg | 2023-06-16 21:13:16 | 2023-06-16 21:13:16 |

-this table named 'activities' the first column is about 'id' and the second column is about 'text' and the third column is about 'description' and the fourth column is about 'image' .


-this table named services the first column is about 'id' and the second column is about 'text' and the third column is about 'image' .


-this table named 'contact' the first column is about 'id' and the second column is about 'text' and the third column is about 'image'.

# The Part of Backend

## ➢ What is Backend ?

- In web development, the backend refers to the server-side of a website or web application. It is the part of the site that users do not see, but which handles all of the essential tasks such as processing data, storing information, and responding to user requests.

- Backend developers use programming languages such as Java, Python, PHP, and Ruby to build and maintain the backend of websites and web applications. They are responsible for tasks such as:

  - ✓ Designing and developing databases.
  - ✓ Creating and managing APIs.
  - ✓ Writing code to process and store data.
  - ✓ Ensuring the security of the backend.
  - ✓ Troubleshooting and debugging backend problems.

- Backend developers work closely with frontend developers, who are responsible for the visual aspects of a website or web application. The two teams must work together to ensure that the frontend and backend of a website or web application are properly integrated and function smoothly.

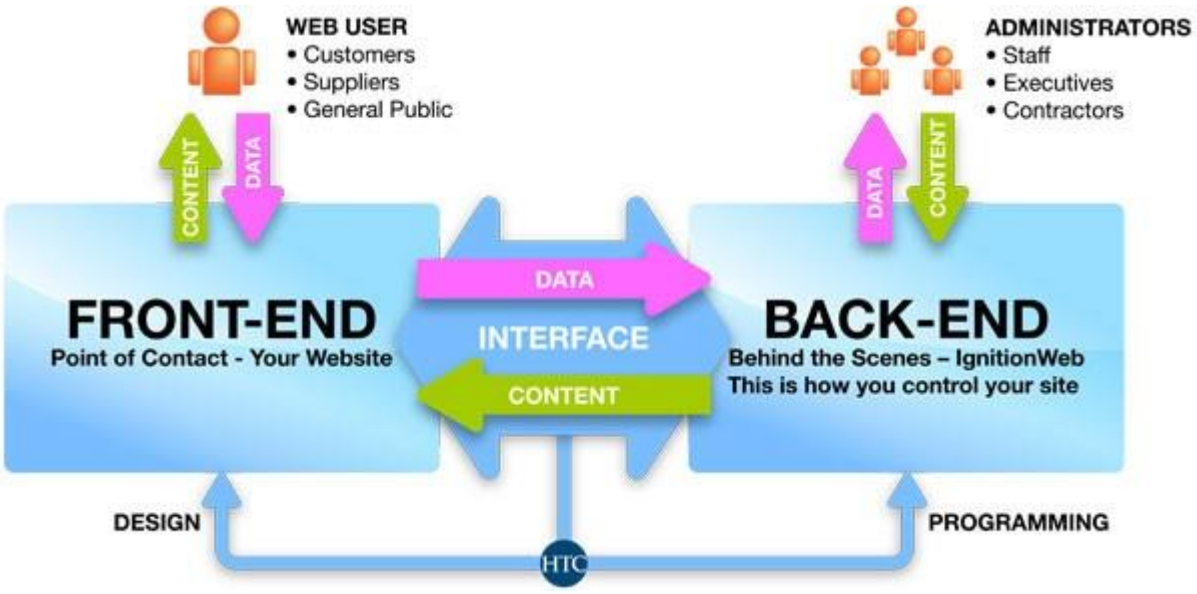
- Here are some examples of backend technologies:

  - ✓ Databases: MySQL, PostgreSQL, MongoDB
  - ✓ Application programming interfaces (APIs): RESTful APIs, SOAP APIs
  - ✓ Web servers: Apache, Nginx, IIS
  - ✓ Programming languages: Java, Python, PHP, Ruby
  - ✓ Frameworks: Django, Rails, Spring Boot

## ➢ What is a Website Backend?

- Backend refers to the server-side of a website or web application. It is the part of the site that users do not see, but which handles all of the essential tasks such as processing data, storing information, and responding to user requests.

- The backend of a website is the place that contains all the data and relevant information that is to be shown to the visitors with the help of a browser. The frontend of a website is merely how the information is presented to the users, and it fetches everything from the backend to display in user browsers. The image below shows this concept visually:

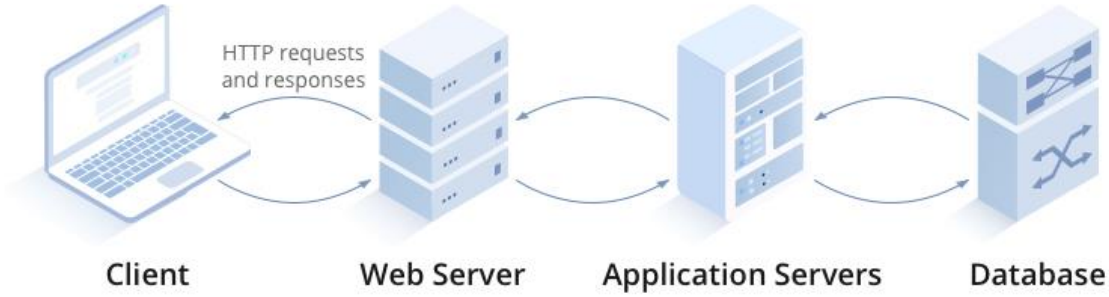


## ➢ How Does Backend Work ?

- So, how does the backend work? It is also known as the server-side of the website, and it requires more than just designing to work properly. Before moving forward, you have to understand what truly is the server-side of a website.

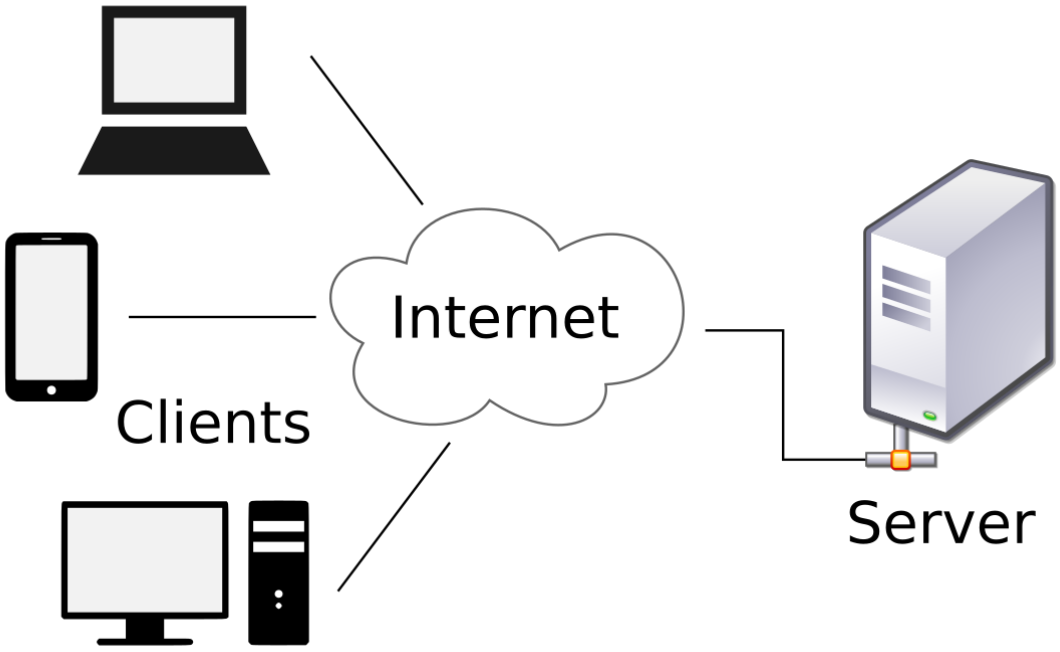
- Here are the three basic components of the server-side of a website. You must understand these before trying to understand how the backend works. These components are actually responsible for handling all the incoming queries from the frontend and sending back the appropriate response and data sets to display on the browser for the users.

    - ✓ A Server
    - ✓ An Application
    - ✓ A Database



## ➢ What Is A Server?

- So, what is a server anyway? You must have heard this term several times in your life as it is one of the most popular terms used in the computer world right now. However, do you understand what a server is and how it functions to help keep everything running smoothly?
- The server's basic purpose is to handle all the incoming requests, and the server is basically a computer that is responsible for handling all the incoming requests with respect to user actions and clicks on the frontend.
- So, when you click something on a website, it prepares a query in a text-based form and sends it back to the server, another computer responsible for handling the incoming queries and sending back the right set of information for your purpose.
- This server machine has all the information in databases, and it is more than capable of handling multiple requests simultaneously. Nowadays, people often use computers that are specifically designed to be used as server computers. However, you can turn any connected computer into a server for your web backend without any issues.

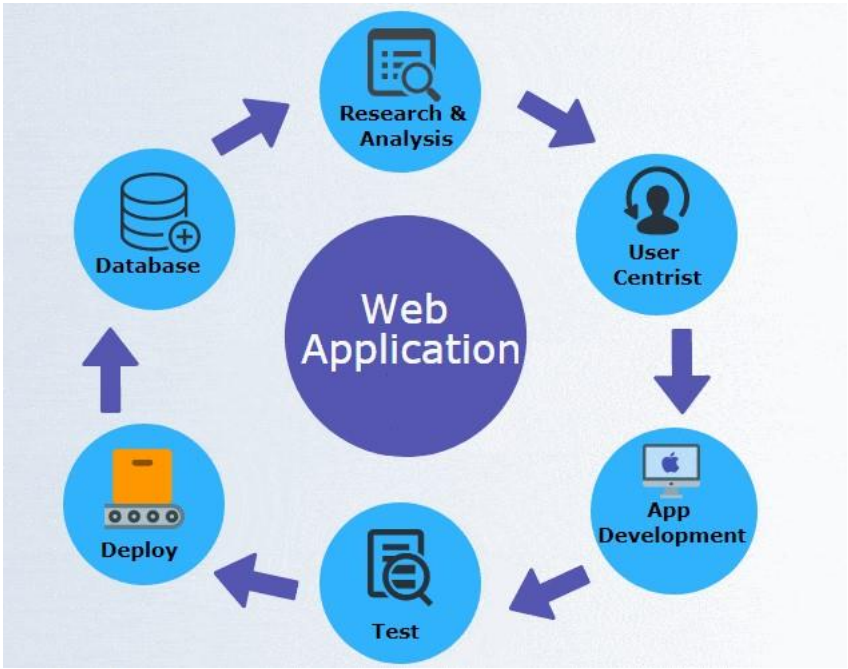


## ➢ What Is An Application?

- The second most important part of the web backend is the application. So, what actually does this app do, and how does it help with the core functions of the server-side mechanism? A server computer can handle requests, but it needs a logical brain that can understand the incoming request and translate it into a set of requirements. Therefore, all the server machines have different applications for different websites responsible for handling HTTPS requests for this server.

- These applications' core function is to understand the logic in text-based commands and respond to that logic by sending back the right set of information. These applications serve as intermediaries between the frontend and the server-side servers.

- These translate the received incoming requests into a set of requirements and communicate the requirements with the server itself. Upon receiving approval and authentication from the server, these applications then get the requested data from the database and send it back to the frontend in an appropriate form.

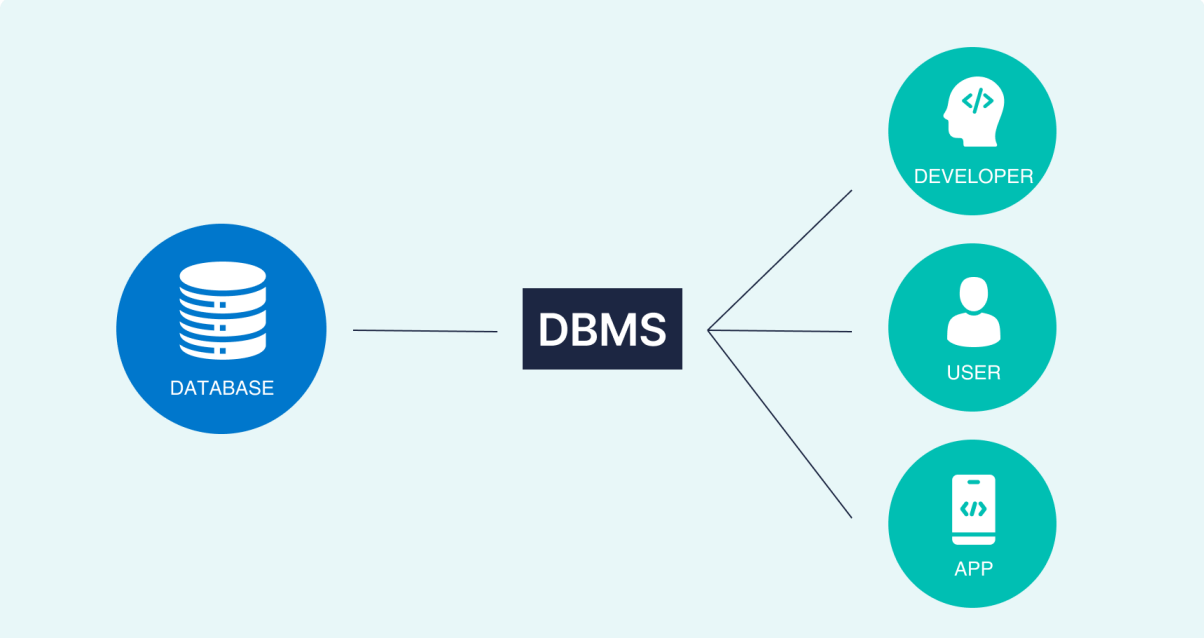


## ➢ What Is A Database?

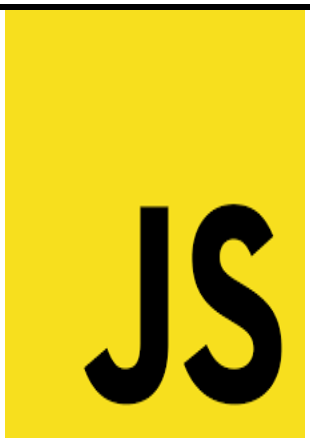
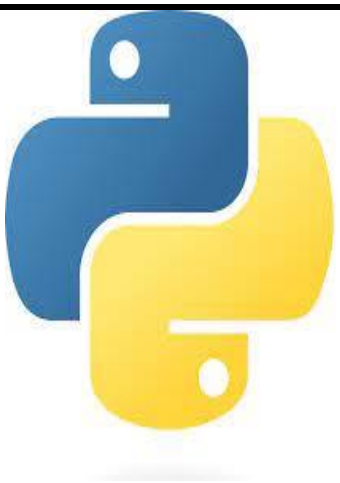
- The database is yet another popular term that you must have heard at least hundreds of times in your life in the recent past. However, do you understand what a database is and its core functionalities as a server-side object?

- A database is a collection of all the different information stored on a machine or a server computer. A database consists of different tables with appropriate column names. These tables contain information that is used to fill out the different boxes and components in the frontend of any website.

- Databases contain all the information ever received for any website or application, and these are hubs of broken-down pieces of information that are later used for various purposes. A database is also known as a building block for any web application or website as this helps provide a platform to save data in a persistent way to the server-side machine memory.



## ➢ The popular languages in Backend Development

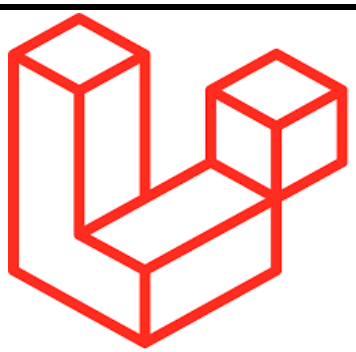
| Language | Description | Logo |
|---|---|---|
| JavaScript | • is a popular choice for backend development because it is a versatile language that can be used for both client-side and server-side development. JavaScript is also very popular among developers, which means that there is a large community of support available.<br>• is a versatile programming language that is commonly used in both frontend and backend development. It is known for its high performance, scalability, and the Node.js runtime environment, which allows developers to run JavaScript on the server-side. |  |
| Python | • is another popular choice for backend development. Python is a general-purpose language that is known for its simplicity and readability. Python is also a very popular language for data science, which makes it a good choice for backend development of applications that involve data processing.<br>• is a versatile programming language that is known for its simplicity, readability, and vast library of modules and frameworks. It is used in a wide range of applications, including web development, data analysis, and machine learning. |  |
| PHP | • is a veteran language that is still widely used for backend development. PHP is a server-side language that is known for its flexibility and ease of use. PHP is also a very popular language for developing web applications.<br>• is a popular server-side scripting language that is commonly used in web development. It is known for its ease of use, flexibility, and compatibility with many web servers and databases. |  |

| | | |
|---|---|---|
| Java | • is a powerful language that is often used for backend development. Java is a compiled language, which means that it is more efficient than interpreted languages like JavaScript. Java is also a very portable language, which means that it can be run on a variety of platforms. |  |
| Ruby | • is a dynamic language that is known for its expressiveness and simplicity. Ruby is a good choice for backend development because it is easy to learn and use. Ruby is also a very popular language for developing web applications. |  |
| Go | • is a newer language that is gaining popularity for backend development. Go is a compiled language that is known for its performance and scalability. Go is also a very concise language, which makes it easy to write and maintain code. |  |

✓ **Here is a table that summarizes the pros and cons of some of the most popular backend programming languages:**

| Language | Pros | Cons |
|---|---|---|
| JavaScript | Versatile, popular, large community | Not as efficient as compiled languages |
| Python | Simple, readable, popular for data science | Not as performant as compiled languages |
| PHP | Flexible, easy to use, popular for web development | Not as secure as some other languages |
| Java | Powerful, portable, efficient | More complex to learn than some other languages |
| Ruby | Expressive, simple, popular for web development | Not as performant as some other languages |
| Go | Performance, scalability, concise | Less mature than some other languages |

## ➢ Frameworks that used in Backend Development

| Framework | Description | Logo |
|---|---|---|
| Laravel | • is a PHP framework that is known for its elegance and flexibility. It is a good choice for developing web applications that require a lot of customizability.<br>• is a popular PHP-based framework for building web applications. It provides developers with a range of tools and features, including an ORM system, a templating engine, and built-in authentication. |  |
| Django | • is a Python framework that is known for its simplicity and scalability. It is a good choice for developing large and complex web applications.<br>• is a popular Python-based framework for building web applications. It provides developers with a range of tools and features, including an Object-Relational Mapping (ORM) system, built-in authentication, and a templating engine. |  |
| Ruby on Rails | • is a Ruby framework that is known for its productivity and simplicity. It is a good choice for developing web applications that require a rapid development cycle.<br>• is a popular Ruby-based framework for building web applications. It provides developers with a range of tools and features, including an ORM system, built-in authentication, and a templating engine. |  |
| Spring Boot | • is a Java framework that is known for its performance and scalability. It is a good choice for developing web applications that need to be highly performant.<br>• is a popular Java-based framework for building web applications and services. |  |
| Express | • is a JavaScript framework that is known for its simplicity and performance. It is a good choice for developing web applications that need to be lightweight and efficient.<br>• is a popular Node.js-based framework for building web applications and APIs. It provides developers with a range of tools and features, including middleware support, routing, and a templating engine. |  |

| | | |
|---|---|---|
| ASP.NET Core | • ASP.NET Core is a .NET framework that is known for its performance and scalability. It is a good choice for developing web applications that need to be highly performant. |  |



✓ **Here is a table that summarizes the pros and cons of some of the most popular backend frameworks:**

| Framework | Pros | Cons |
|---|---|---|
| Laravel | Elegant, flexible, active community | Can be complex for beginners |
| Django | Simple, scalable, well-documented | Not as flexible as some other frameworks |
| Ruby on Rails | Productive, simple, good for rapid development | Not as performant as some other frameworks |
| Spring Boot | Performance, scalability, well-documented | Can be complex for beginners |
| Express | Simple, lightweight, performant | Not as feature-rich as some other frameworks |
| ASP.NET Core | Performance, scalability, .NET ecosystem | Can be complex for beginners |

➢ **In this project we use PHP language and Laravel framework:**

➢ **The following are the steps through which this project was implemented:**

# 1- About Controller

```php
public function insert(Request $request){
    if (about::all()->count()==0) {
        $insert=new about();
        $image=$request->file('image');
        $image_name=time().$image->getClientOriginalName();
        $image->move('images_about/',$image_name);

        $insert->text=$request->text;
        $insert->image=$image_name;
        $insert->save();
        return back();

    }
    else{

        $insert=about::all()->first();
        unlink('images_about/'.$insert->image);

        $image=$request->file('image');
        $image_name=time().$image->getClientOriginalName();
        $image->move('images_about/',$image_name);

        $insert->text=$request->text;
        $insert->image=$image_name;

        $insert->save();
        return back();
```

- **The code is a Laravel controller function that inserts a new about record into the database.**
- **The function first checks if there are any about records in the database.**
- **If there are no records, then a new about record is created and the text and image values from the request are saved to the record.**
- **The image is uploaded to the (images_about) directory. The record is then saved to the database and the function returns back.**

# 2- Activitys Controller

```php
class ActivitysController extends Controller
{


    public function insert(Request $request){
        $insert=new activitys();
        $image=$request->file('image');
        $image_name=time().$image->getClientOriginalName();
        $image->move('images_activitys/',$image_name);

        ////////////////////

        $insert->image=$image_name;
        $insert->name=$request->name;
        $insert->text=$request->text;
        $insert->save();
        return back();
    }
    public function get_data_web (){
        $data= activitys::all();
        return view('activitys')->with('data',$data);
```

- **The function first creates a new activity record and assigns the image, name, and text values from the request to the record.**
- **The image is uploaded to the images_activitys directory. The record is then saved to the database and the function returns back.**

```php
    public function get_data_web (){
        $data= activitys::all();
        return view('activitys')->with('data',$data);

    }

    public function delete_activity(Request $request){
        $item=activitys::find($request->id);
        unlink('images_activitys/'.$item->image);
        $item->delete();
        return back();

    }

    public function get_data (){

        return activitys::all();

    }

}
```

- **The first function, get_data_web(), retrieves all of the activity records and passes them to the activitys view.**
- **The second function, delete_activity(), deletes the activity record with the specified ID.**

# 3- Contact Controller

```php
public function insert(Request $request){
    if (contact::all()->count()==0) {
        $insert=new contact();
        $image=$request->file('image');
        $image_name=time().$image->getClientOriginalName();
        $image->move('images_contact/',$image_name);

        $insert->text=$request->text;
        $insert->image=$image_name;
        $insert->save();
        return back();

    }
    else{

        $insert=contact::all()->first();
        unlink('images_contact/'.$insert->image);

        $image=$request->file('image');
        $image_name=time().$image->getClientOriginalName();
        $image->move('images_contact/',$image_name);

        $insert->text=$request->text;
        $insert->image=$image_name;

        $insert->save();
        return back();
```

- **The function first checks if there are any contact records in the database.**
- **If there are no records, then a new contact record is created and the text and image values from the request are saved to the record.**
- **The image is uploaded to the images_contact directory. The record is then saved to the database and the function returns back.**

```php
public function get_data(){
    return contact::all();
}
public function get_data_web(){
    $data=contact::all();
    return view('contact')->with('data',$data);
```

- **The first function, get_data(), simply returns all of the contact records.**
- **The second function, get_data_web(), retrieves all of the contact records and passes them to the contact view.**

# 4- Header Controller:

```
class HeaderController extends Controller
{
    //
    public function insert(Request $request){
        if (header::all()->count()==0) {
            $insert=new header();
            $insert->text=$request->text;
            $insert->head=$request->head;
            $insert->save();
            return back();

        }
        else{
            $insert=header::all()->first();
            $insert->text=$request->text;
            $insert->head=$request->head;
            $insert->save();
            return back();


        }
```

- **The function first checks if there are any header records in the database.**
- **If there are no records, then a new header record is created and the text and head values from the request are saved to the record.**
- **The record is then saved to the database and the function returns back.**

```
    public function get_data(){
        return header::all();
    }
    public function get_data_web(){
        $data=header::all();
        return view('header')->with('data',$data);
    }

}
```

- **The first function, [ get_data() ], simply returns all of the header records.**
- **The second function, [ get_data_web() ], retrieves all of the header records and passes them to the header view.**

# 5- Home Text Controller

```
class HomeTextController extends Controller
{
    //
        public function insert(Request $request){
        if (home_text::all()->count()==0) {
            $insert=new home_text();
            $insert->text=$request->text;

            $insert->save();
            return back();

        }
        else{
            $insert=home_text::all()->first();
            $insert->text=$request->text;
            $insert->save();
            return back();
```

- **The function first checks if there are any home_text records in the database.**
- **If there are no records, then a new home_text record is created and the text value from the request is saved to the record.**
- **The record is then saved to the database and the function returns back.**

```
public function get_data(){
    return home_text::all();
}
public function get_data_web(){
    $data=home_text::all();
    return view('home_text')->with('data',$data);
```

- **The first function, get_data(), simply returns all of the home_text records.**
- **The second function, get_data_web(), retrieves all of the home_text records and passes them to the home_text view.**

## 6- Main Activity Controller

```php
public function insert(Request $request){
  if (main_activity::all()->count()==0) {
      $insert=new main_activity();
      $image=$request->file('image');
      $image_name=time().$image->getClientOriginalName();
      $image->move('images_main_activity/',$image_name);

      $insert->text=$request->text;
      $insert->description=$request->description;
      $insert->image=$image_name;
      $insert->save();
      return back();

  }
  else{

      $insert=main_activity::all()->first();
      unlink('images_main_activity/'.$insert->image);

      $image=$request->file('image');
      $image_name=time().$image->getClientOriginalName();
      $image->move('images_main_activity/',$image_name);

      $insert->text=$request->text;
      $insert->image=$image_name;
      $insert->description=$request->description;

      $insert->save();
      return back();
```

- **The function first checks to see if there are any existing main_activity records.**
- **If there are no existing records, then the function creates a new record and saves it to the database.**
- **If there are existing records, then the function updates the first record in the database with the new data.**
- **The function first gets the image file from the request.**
- **The image file is then moved to the images_main_activity/ directory and the file name is generated using the current time and the original file name.**
- **The text and description data from the request are then set on the new or updated main_activity record. Finally, the record is saved to the database.**

# 7- Main Service Controller

```php
public function insert(Request $request){
    if (main_service::all()->count()==0) {
        $insert=new main_service();
        $image=$request->file('image');
        $image_name=time().$image->getClientOriginalName();
        $image->move('images_main_service/',$image_name);

        $insert->text=$request->text;
        $insert->image=$image_name;
        $insert->save();
        return back();

    }
    else{

        $insert=main_service::all()->first();
        unlink('images_main_service/'.$insert->image);

        $image=$request->file('image');
        $image_name=time().$image->getClientOriginalName();
        $image->move('images_main_service/',$image_name);

        $insert->text=$request->text;
        $insert->image=$image_name;

        $insert->save();
        return back();
```

- **The function first checks to see if there are any existing main_service records.**
- **If there are no existing records, then the function creates a new record and saves it to the database.**
- **If there are existing records, then the function updates the first record in the database with the new data.**
- **The function first gets the image file from the request.**
- **The image file is then moved to the images_main_service/ directory and the file name is generated using the current time and the original file name.**
- **The text data from the request is then set on the new or updated main_service record. Finally, the record is saved to the database.**

# 8- Member Controller

```php
class MemberController extends Controller
{
    public function insert(Request $request){
        $insert=new member();
        $insert->name=$request->name;
        $insert->text=$request->text;
        $insert->save();
        return back();
    }

    public function get_data(){
        return member::all();
    }
    public function get_data_web(){
        $data=member::all();
        return view('member')->with('data',$data);
    }

    public function delete_member(Request $request){
        $item=member::find($request->id);

        $item->delete();
        return back();
```

- **insert(): This function inserts a new member record into the database.**
- **get_data(): This function returns all of the member records from the database.**
- **get_data_web(): This function returns all of the member records from the database and renders them as a view.**
- **delete_member(): This function deletes a member record from the database.**

# 9- Message Controller

```php
class MessageController extends Controller
{
    //
    public function insert(Request $request){
        $insert=new message();
        $insert->name=$request->name;
        $insert->email=$request->email;
        $insert->phone=$request->phone;
        $insert->message=$request->message;
        $insert->save();
        return 'done';

    }
    public function get_data_web(){
        $data=message::all();
        return view('message')->with('data',$data);
    }
        public function delete_message(Request $request){
        $item=message::find($request->id);
        $item->delete();
        return back();
```

- **The function first creates a new message record and sets the name, email, phone, and message fields from the request. Then, the record is saved to the database.**
- **The function first gets all of the message records from the database.**
- **Then, it renders the message view, passing the message records to the view as a variable.**
- **The code could be made more modular by using functions to perform specific tasks, such as getting the message records from the database or rendering the message view.**

# 10- Our Services Controller

```php
class OurServicesController extends Controller
{
    //
    public function insert(Request $request){
        $insert=new our_services();
        $insert->main_text=$request->main_text;
        $insert->name_item=$request->name_item;
        $insert->text_item=$request->text_item;
        $insert->save();
        return back();
    }

    public function insert_item(Request $request){
        $insert=new our_services();
        $main=our_services::all()->first();
        $insert->main_text=$main->main_text;
        $insert->name_item=$request->name_item;
        $insert->text_item=$request->text_item;
        $insert->save();
        return back();
```

- **insert(): This function inserts a new our_services record into the database.**
- **insert_item(): This function inserts a new our_services item record into the database.**
- **The insert() function first creates a new our_services record and sets the main_text, name_item, and text_item fields from the request. Then, the record is saved to the database.**

```php
public function get_data(){
    return our_services::all();
}
public function get_data_web(){
    $data=our_services::all();
    return view('our_services')->with('data',$data);
}

public function delete_our_services(Request $request){
    $item=our_services::find($request->id);

    $item->delete();
    return back();

}

public function get_data_web_item(){
    $data=our_services::all();
    return view('item_our_services')->with('data',$data);
```

- **get_data(): This function returns all of the our_services records from the database.**
- **get_data_web(): This function returns all of the our_services records from the database and renders them as a view.**
- **delete_our_services(): This function deletes a our_services record from the database.**

# 11- Our Team Controller

```php
public function insert(Request $request){
    if (our_team::all()->count()==0) {
        $insert=new our_team();
        $image=$request->file('image');
        $image_name=time().$image->getClientOriginalName();
        $image->move('images_team/',$image_name);


        $insert->text=$request->text;
        $insert->image=$image_name;

        $insert->save();
        return back();

    }
    else{
        $insert=our_team::all()->first();
        unlink('images_team/'.$insert->image);

        $image=$request->file('image');
        $image_name=time().$image->getClientOriginalName();
        $image->move('images_team/',$image_name);

        $insert->text=$request->text;
        $insert->image=$image_name;

        $insert->save();
        return back();
```

- **The function first checks to see if there are any existing our_team records.**
- **If there are no existing records, then the function creates a new record and saves it to the database.**
- **If there are existing records, then the function updates the first record in the database with the new data.**

```php
public function get_data(){
    return our_team::all();
}
public function get_data_web(){
    $data=our_team::all();
    return view('our_team')->with('data',$data);
```

- **The function first gets all of the our_team records from the database.**
- **Then, it renders the our_team view, passing the our_team records to the view as a variable.**

# Chapter five

## Conclusion and future work

## 5.1- Conclusion :

After the diligent and continuous work on developing a website for the Youth Care Department at the College of Computer and Information Sciences, it can be concluded that the desired objectives of this project have been successfully achieved. The website serves as an effective platform to provide the necessary support and guidance for the college's youth, and it offers various services and activities that cater to their needs and interests. The development process included thorough planning, design, and implementation phases, which resulted in a user-friendly and visually appealing website that meets the highest standards of quality and professionalism. Overall, the project has been a valuable learning experience for all involved parties, and it represents a significant contribution to the college's efforts to enhance the well-being and development of its youth community.

The main objective of the website was to create a platform that would provide support and guidance for the youth community at the College of Computer and Information Sciences. The website offers a wide range of services and activities that cater to the needs and interests of the college's youth, including academic support, career guidance, personal development resources, and social events.

To achieve this objective, the development process was divided into several phases, including planning, design, and implementation. During the planning phase, the project team conducted extensive research on the needs and preferences of the college's youth community, as well as the best practices in website development. This research informed the design phase, during which the team created a user-friendly and visually appealing website that meets the highest standards of quality and professionalism.

The implementation phase involved the actual development of the website, including the creation of various features and functionalities that would facilitate the delivery of the desired services and activities. These features include an events calendar, a resource library, an online chat service, and a feedback mechanism, among others.

Throughout the development process, the project team collaborated closely with the Youth Care Department at the College of Computer and Information Sciences to ensure that the website meets the needs and expectations of the college's youth community. The end result is a website that serves as an effective platform for delivering support and guidance to the college's youth, and that represents a significant contribution to the college's efforts to enhance their well-being and development.

Overall, the project was a valuable learning experience for all involved parties, including the project team, the Youth Care Department, and the college's youth community. It demonstrates the importance of collaboration, planning, and attention to detail in the development of effective websites, and it serves as a model for future projects in this area.

# future work :

A graduation project on youth care in the Faculty of Computers and Information can include several future enhancements :

1.One of these enhancements is enabling students to log in using their university email

2.allowing them to register for various activities within the department through the website.

3. make it easier for students to conduct research on social solidarity

4. Additionally, facilitating social networking and communication with supervisors can be made easier.

5.Furthermore, students should be able to express their opinions and suggestions for improving the website.

## 5.2- Technical Tools

- Software Ideas Modeler Standard Portable
- Visual Studio Code
- XAMPP
- PHP/MYSQL
- Local Host php

# Chapter Six

## References

- Systems analysis and design /Alan Dennis, Barbara Haley Wixom, Roberta M. Roth. –5th ed.
- Systems Analysis and Design, 11th Edition, by Kenneth E. Kendall and Julie E. Kendall.
- Web Design with HTML, CSS, JavaScript and jQuery Set, by Jon Duckett.
- Web Database Applications with PHP & MySQL, 2nd Edition, by Hugh E. Williams and David Lane
- https://www.canva.com/graphs/flowcharts/
- http://www.fci2.zu.edu.eg/YCActs.aspx/
- https://www.facebook.com/profile.php?id=100088928495224&mibextid=ZbWKwL
- https://laracasts.com/series/laravel-6-from-scratch ↗
- https://www.tutorialspoint.com/laravel/index.htm