BAHRIA UNIVERSITY (KARACHI CAMPUS)

Software Design & Architecture (SEN-221)

ASSIGNMENT # 1 – Spring 2022

Based on: CLO-2

Class: **BSE-4B** Submission Deadline: **03rd April 22**

Course Instructor: **ENGR. MAJID KALEEM** Max Marks: **05**

1. Suppose you must design & develop an online LMS System. You may consider various architectural and design style to address this situation. Keep LMS in mind, describe (separately) each of the architectural styles given below with respect to the following questions:

   1. ***What*** are they?
   2. ***Where*** are they used?
   3. ***Why*** are they used?
   4. ***How*** are they used / implemented?

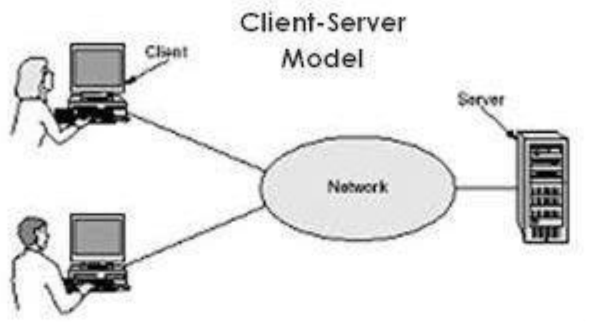
   a) Client/Server Architecture
   b) Component-Based Architecture
   c) MVC Architecture
   d) Cloud Application Architecture

SOLUTION

a) CLIENT/SERVER ARCHITECTURE

WHAT?

It is a paradigm of network computing. It is a shared network in which various devices are connected remotely to a centralized system through which it transmits and receives multiple requests. Workstations, servers, and networking stations are the three main components. Client-server architecture is also known as a networking computer paradigm or client-server network.



WHERE?

The client-server architecture is best suited for applications that require a separation of concerns or abstraction of concerns between the client and the server; it is designed for systems that demand strong interoperability. The client-server architectural style helps application improve in scalability and performance.

There is an E-book system at our library. So, there is a primary server where all E books are uploaded, and then each device in the institution may access them over the internet. In this case, a single server is linked to all client devices and delivers eBooks in response to client requests.

WHY?

We utilize client server because you require a major storage center from which you can simply retrieve data when utilizing an LMS system like this. It also aids in the secure storing of data and the prevention of data loss.

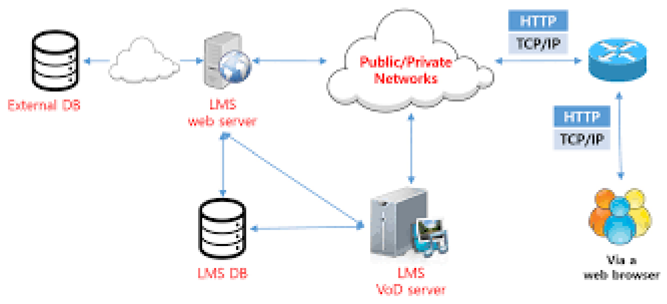
HOW?

Many clients request and receive service from a centralized server in a client-server architecture **(host computer)**. Client computers provide an interface that allows a computer user to request server services and view the results returned by the server. Servers wait for requests from clients to arrive before responding. Clients should not need to be aware of the specifics of the system (i.e., the hardware and software) that is providing the service if a server provides a standardized transparent interface. Clients are often found at workstations or on personal computers, whereas servers are typically found elsewhere on the network, on more powerful devices. This computing architecture is especially useful when clients and servers each have separate duties to do on a regular basis. Data processing at an E-commerce store.

<u>EXAMPLE</u>:

- Email
- Network printing
- World Wide Web

These are examples of computer applications that employ the client–server concept. Protocols are a collection of rules that end points of a telecommunication connection use to communicate with one another. TCP/IP, HTTP, FTP, and other protocols are examples.



b) <u>COMPONENT-BASED ARCHITECTURE</u>

<u>WHAT?</u>

Component-based architecture is a sort of application architecture made up of reusable, modular, and independent building parts known as components. Developers integrate, reuse, and version these items when constructing an app using component-based architectural concepts, saving a lot of effort over building every inch of an app from scratch. It offers a higher degree of abstraction and separates the issue into sub-problems, each with its own set of component partitions.

<u>WHERE?</u>

Component-based design can be kept up to date without having to rebuild it from the scratch. Companies with massive, monolithic codebases will benefit from component-based design. A monolith becomes software building blocks when components are used. These components may also be mixed and matched, reused, and versioned.

There are several components in the LMS, like as notes and tests. Assignments, and so forth. During the pandemic, Bahria refined his examination section on an existing system so that if a component-based architecture existed, it would be simple to deploy.

<u>WHY?</u>

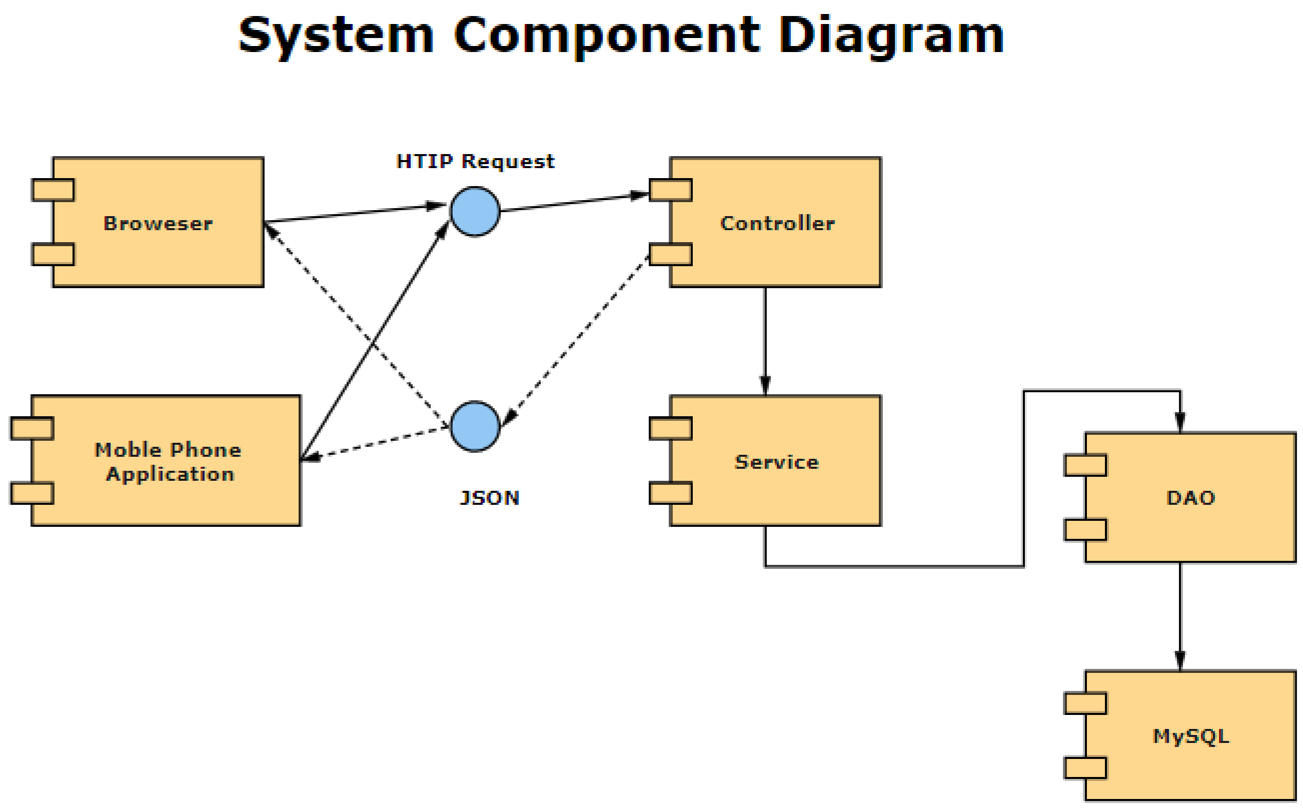
The major goal of component-based architecture is to ensure that components may be reused. A component is a reusable and self-deployable binary unit that contains the functionality and behaviors of a software element. Component frameworks such as JavaBean, CORBA, .NET, web services, and grid services are all common. Graphic JavaBean components, MS ActiveX components, and COM components, all of which can be reused by simply dragging and dropping, are commonly utilized in local desktop GUI application design.

HOW?

A component-level design can be represented using a graphical, tabular, or text-based intermediate representation that can be converted into source code. To assist us avoid introducing mistakes, the design of data structures, interfaces, and algorithms should follow well-established rules. The software system is broken down into component components that are reusable, coherent, and encapsulated. Each component has its own interface that describes the ports that are required and given; each component hides the details of its implementation. A component should be able to be expanded without requiring internal code or design changes to the component's current pieces. Components that rely on abstractions do not rely on other tangible components, increasing the challenge of expendability. Connectors link components together, defining and governing their interactions. The interfaces of the components define the type of interaction.

EXAMPLES

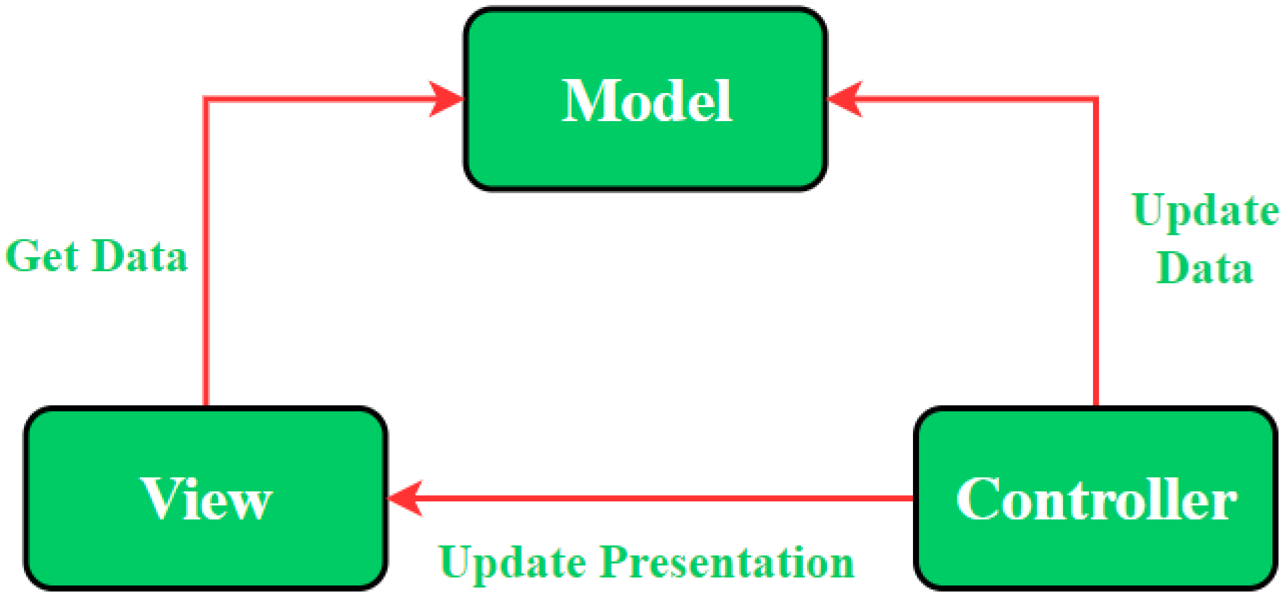
Layered architecture can be wrapped inside a component in a component-based architecture. A payment service (component), for example, can have three layers: controller, service, and model. In this example, the layered architecture is more of an implementation detail than a structure.



c) MVC ARCHITECTURE

WHAT?

The Model-View-Controller (MVC) architectural pattern divides an application into three logical components: model, view, and controller. Each of these components is designed to handle certain parts of application development. MVC is a popular industry-standard web development framework for developing scalable and flexible projects. It stresses the distinction between the business logic and the appearance of the software. This "separation of concerns" allows for a better division of labor and improved maintenance.



WHERE?

The MVC design pattern is a fantastic way to create online apps. Several businesses are choosing to construct such applications using the MVC design. It becomes quite simple to split and arrange the logic of web apps into large-scale applications because the MVC design is compatible with JavaScript and its frameworks, it's no surprise that it also supports Asynchronous Method Invocation (AMI), which allows developers to create web applications that load quicker. It implies MVC apps may be configured to function with PDF files, site-specific browsers, and desktop widgets as well.

WHY?

It aids in the creation of applications with many characteristics like as input logic, business logic, and user interface logic. The pattern indicates where each type of logic in the application should be placed.

1. UI logic belongs in the view in this case.
2. Input logic should be handled by the controller.
3. Business logic should be included in the model.

It is used because

- Easily Modifiable
- Faster Development Process
- Easy planning and maintenance
- SEO-Friendly Platform

HOW?

MVC consist of three parts

- Model
- View
- Controller

MODEL

It would contain all the data logic that we would build in the LMS, for example (submission of paper within given time, managing of submitted documents of assignment, handling of traffic users).

VIEW

In MVC, a view is a user interface. View gives the user access to model data and allows them to edit it. In ASP.NET MVC, the view is composed of HTML, CSS that facilitates communication with the model and controller. We will determine how our LMS system will be shown in this section.

CONTROLLER

The user's request is handled by the controller. The user typically uses the view to issue an HTTP request, which is processed by the controller. As a response, the controller processes the request and returns the relevant view.

EXAMPLE

We'll use some logic such that once a certain amount of time has passed, the student won't be allowed to submit his assignment or the paper in the paper part.

EXAMPLE

Car driving mechanism is another example of the MVC model. Every car consists of three main parts. View is User interface: (Gear lever, panels, steering wheel, brake, etc.) Controller- Mechanism (Engine) Model- Storage (Petrol or Diesel tank).

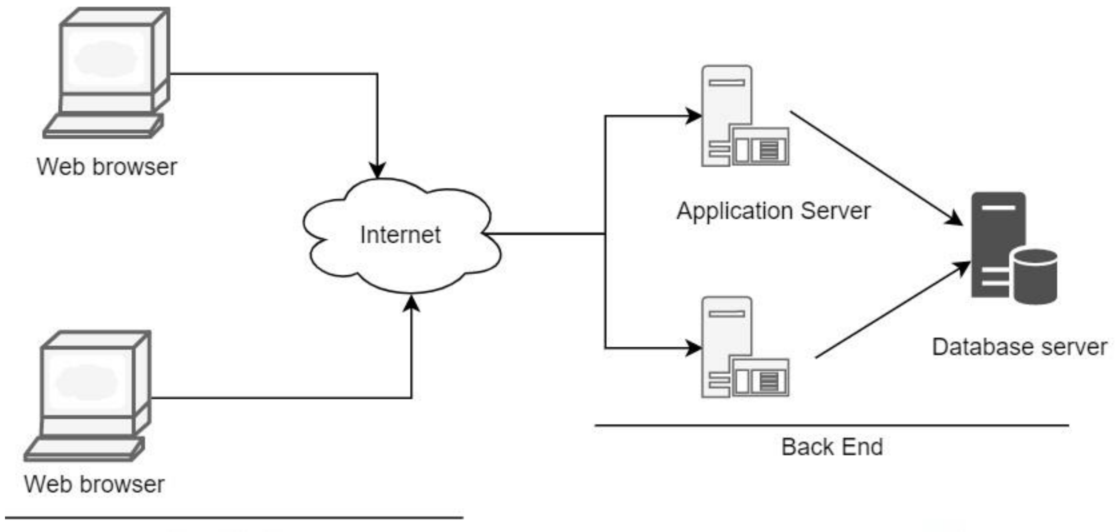
d) CLOUD-APPLICATION ARCHITECTURE

WHAT?

The way technological components come together to form a cloud, in which resources are pooled and shared over a network, is known as cloud architecture. A cloud architecture consists of the following elements:

- A platform for front-end development (the client or device used to access the cloud)
- A platform for the backend (servers and storage)
- A delivery paradigm based on the cloud

These technologies work together to build a cloud computing infrastructure on which applications may operate, allowing end-users to take use of cloud resources.



WHERE?

Cloud computing is a good fit for anything that involves storing and processing large amounts of data at rapid speeds and necessitates more storage and computing capability than most businesses can or want to buy and implement on-premises. Here are several examples:

- Analysis of large amounts of data
- The Internet of Things (IoT).
- Machine learning and deep learning applications are examples of artificial intelligence.

EXAMPLE

- Microsoft Azure.
- Amazon Web Services.
- Google Cloud.
- Alibaba Cloud.
- IBM Cloud.

WHY?

The adoption of a cloud architecture is motivated by a variety of factors. Some of them are following

- Improve the speed with which new apps are released.
- To update apps and expedite digital transformation, use cloud-native architecture like Kubernetes.
- Ensure that the most recent regulations are followed.
- Increase resource transparency to save expenses and avoid data breaches.
- Allow for speedier resource provisioning.
- As business demands evolve, use a hybrid cloud architecture to allow real-time scaling for apps.
- Consistently meet service goals
- Use cloud reference architecture to have a better understanding of IT expenditure trends and cloud use.

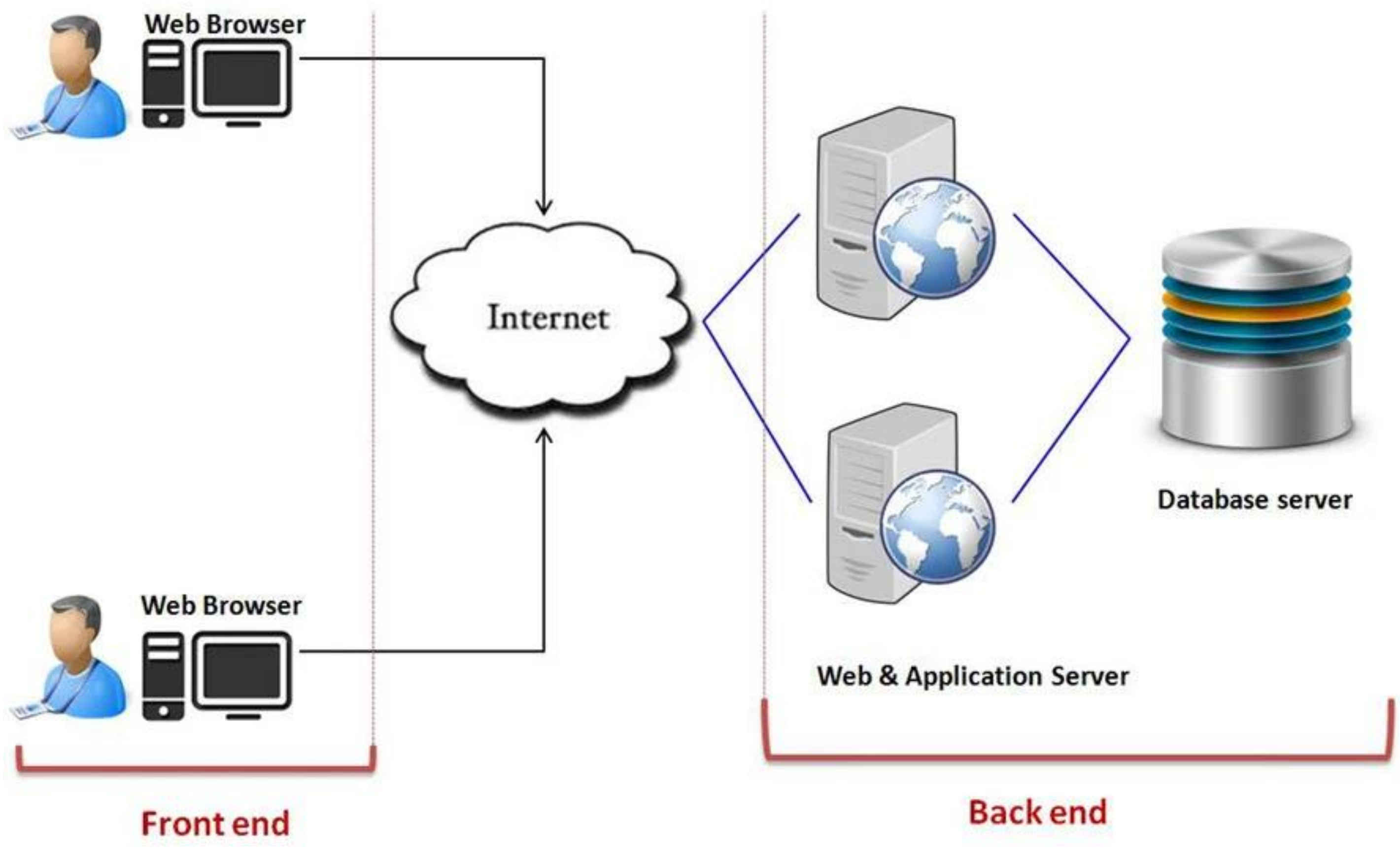
HOW?

The front-end and backend of Cloud Computing Architecture are separated into two components. Both the front-end and the back end of a website are important. use the internet or a network to communicate.
There is a database in our organization. The database oversees data storage and security, as well as serving as the backend. This database is maintained on the internet and serves as storage for the LMS. Here, the LMS serves as

the front end, allowing you to perform functions.

<u>EXAMPLE</u>

Any application or platform that a customer desires to use might be considered an application. A Cloud Services handles which sort of service you use based on the needs of the customer. Google Apps and Salesforce are two examples. Dropbox, Slack, HubSpot, and Cisco WebEx are just a few examples.



REFERENCES

- https://www.simplilearn.com/tutorials/cloud-computing-tutorial/cloud-computing-architecture
- https://msatechnosoft.in/blog/types-of-client-server-architecture/
- https://www.thecrazyprogrammer.com/2021/03/client-server-architecture.html
- https://developer.mozilla.org/en-US/docs/Glossary/MVC
- https://www.tutorialspoint.com/software_architecture_design/component_based_architecture.htm
- https://www.guru99.com/mvc-tutorial.html
- https://cs.uwaterloo.ca/—m2nagapp/courses/CS446/1195/Arch Design Activity/ClientServer.pdf
- https://cio-wiki.org/wiki/Client
- https://u"vvw.britannica.com/technology/client-server-architecture
- https://teachcomputerscience.com/client-server-architecture/
- http://digitalthinkerhelp.com/what-is-client-server-architecture-diagram-types-examples-
- https://wvvw.outsystems.com/blog/posts/component-architecture/
- https://www.tutorialspoint.com/software architecture design/component
- https://www.perforce.com/blog/vcs/component-based-development
- https://en.wikipedia.org/wiki.Com