

**TECHNICAL DEPARTMENT**

Name…FARAI MAKOMBORERO

Surname…MATSIKA

Registration number……T2128646Y

Programme: Software Engineering Class A (S:E)

Module: EAD 2

Assignment number: One

Due date…04/06/2023

Mark

Lecturer’s signature…………….

Question 1.

1. What are different components of WSDL? [6]
2. Message.
3. Port type.
4. Binding
5. Types
6. Operation.
7. Soap: binding.
8. Service.
9. State 5 advantages of REST Web Services? [5]
10. They are easier to build and learn as they are simple.
11. They are fast and they consume less bandwidth and resources.
12. They support various data formats such as HTML, XML and many more.
13. They can manage high load through the use of proxy servers and cache.
14. The services can be written in any language and can be run on any platform.
15. State 5 disadvantages of SOAP Web Services? [5]
16. Compared to Rest, the SOAP web services are much complex to code and thereby takes more time to complete.
17. They cannot be setup easily, they also require some expertise during the process.
18. Other lightweight formats are not supported, it only supports XML.
19. The XML format which it uses needs to be parsed for it to be read, and it becomes slow.
20. As SOAP is based on the contract, there is a tight coupling between client and server applications.
21. XML, SOAP, WSDL and UDDI are all web service standards that work together to enable communication between different applications over the internet.

|  |  |  |  |
| --- | --- | --- | --- |
| XML | SOAP | WSDL | UDDI |
| Is a markup language used to encode data. It provides standard ways of representing data that can be easily exchanged between different systems. | Is a messaging protocol used for exchanging structured information between applications. It uses XML as its message format. | Is an XML-based language used for describing the functionality offered by a web-service. | Is a platform -independent registry for businesses to list their web services. It provides a standard way of discovering and locating web services over the internet. |

In brief, XML provides the data format exchanged between applications, SOAP provides the messaging protocol, WSDL describes functionality offered by web service and UDDI provides a registry for discovering web services.

5. Differentiate between .NET Web Services and .NET Remoting? [4]

|  |  |
| --- | --- |
| .NET Web Services | .Net Remoting |
| They work in a stateless environment | Provide support for both stateful and stateless environments. |
| Easy to create and deploy | They are complex to program. |
| They can be accessed over HTTP | They can be accessed over any protocol including TCP, SMTP and others |

6. Discuss the concept of variable variables, as it applies to php? [5]

In php, variable variables is a feature that allows creation of variable names dynamically, based on the value of another variable. They are created by placing the variable name inside another set of curly braces. For example, if we have the variable $b with value of “hello”, we can create a new variable with the name “hello” by doing this: ${$b} =” world”;

That creates a new variable with the name hello which is the value of $b and this variable will have a value of “world”. Sow we now have the variable hello with “world” as its value. Variables can be useful in certain situations like when working with dynamic data. However, they can make code harder to read and debug and there can be security risks if they are not used properly but the bottom line is they are essential and should be used with caution.

Discuss how Transaction-Processing Architectural Pattern subscribes to the principles of good architectural design. [10]

Transaction processing application processes request in a serial sequence. The processing of simple requests involves receiving and routing a requesting to the application program and doing the execution. The front-end program is responsible for gathering input for the requests and the name of the transaction to be executed. This is where the process starts.

Transaction processing architectural pattern subscribes to the principles of good architectural design in the sense that, the transaction handlers are defined as suitable system divisions that can be given to different developers for detailed design and development and this can help to ensure that complex tasks are broken down into smaller units and this suits well with the divide and conquer principle. As the design principles state that there should be reduced coupling, the pattern separates the dispatcher from the handlers clearly and this helps to make changes effectively without affecting many things at once. Moreover, One may add assertion checking in each transaction handle in the dispatcher and this suits really well with the principle of designing defensively. On this pattern, transaction handlers are naturally cohesive as they inter relate in their operations. Handlers with different operations are separated to promote cohesion. They also provide an abstraction called a container which helps the application developers to handle the cumbersomeness of transaction management and low level operating systems.

8. The model view controller architecture is an architectural pattern that encourages improved application organization through separation of concerns. This principle aims to separate a application into district sections, where each section addresses a specific separate issue. It divides the interactive application into three components. It contains underlying classes whose objects are to be viewed and manipulated.:

Model – this only contains data pure application data with no logic describing how it will be presented to the user. It generally contains classes from the domain that may be general and form the application. It is the central component of the pattern and it is the application’s dynamic structure which is independent of the user interface.

View – Contains objects which are used to give an appearance to the data from the model in the user interface. Any representation of information such as a chart, diagram or table is called a view. It presents the model’s data to the user. The view knows how to access the model’s data without knowing the meaning or what the user can do to manipulate it.

Controller – This is the backbone of the pattern usually. It exists between the view and the model and this is where the actual business logic is written. It listens to the events from the view as its input and responds with execution or anything that d=has been requested by the event triggered.

The MVC exhibits layer cohesion as each layer has no idea about what it is attached to. All the three layers reside in different architectural layers.