2h 5010

1.5h part2

what the below concepts are and why and how are they used in an example eCommerce web application (with example)

1. Model View Controller (MVC) design pattern

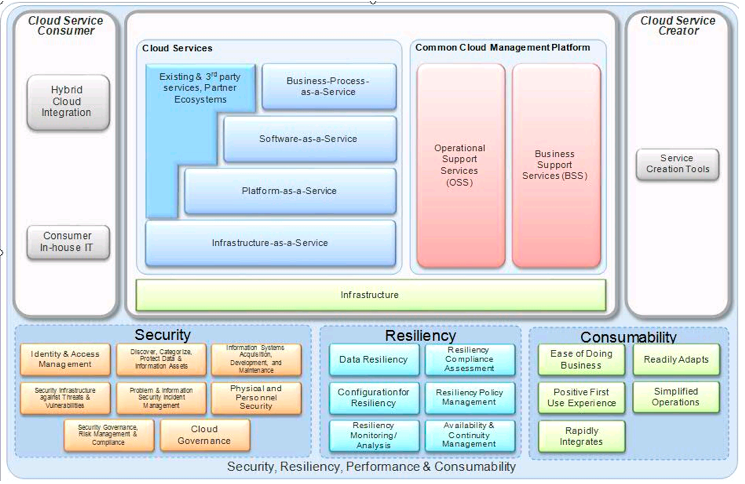
Model view controller (MVC) is a software design pattern used for developing user interfaces that divide the related program logic into three interconnected elements. The goal of MVC definition is to streamline communication among developers.

Model is the central component of the pattern. The controller is an intermediary between the Model and the View. It accepts user input from the View, and converts it to commands for the Model or View.

In the MVC eCommerce web application development, the controller receives all requests for the application and then instructs the model to prepare any information required by the view. The view uses that data prepared by the controller to bring the final output. For an example eCommerce web application (Amazon), MVC can support faster development process, support for asynchronous technique, and has ability to provide multiple Views. In summary, with the help of the MVC model, Amazon web application can be easily developed with lesser expenditure and within less time.

2. Cloud Computing reference Architecture for infrastructure

CCRA is an IBM-defined reference architecture for the cloud computing domain. It is an evolving architecture that is based on real-world input from many cloud implementations. The CCRA defines the basic building blocks—architectural elements and their relationships, which make up the cloud.



For an example eCommerce web application (Amazon), CCRA is used as a blueprint for architecting cloud implementations, driven by functional and non-functional requirements of the respective cloud implementation of eCommerce web application (Amazon). So, it improves the time to capability and reduces the overall IT costs associated with private, public and hybrid cloud models for the eCommerce web application.

A picture containing graphical user interface

Description automatically generated

3. RESTful Web Services

Redbook

Python aws notebook

RESTful APIs enable automated management of storage clouds that offer flexibility to meet the needs beyond standard cloud tools. With these APIs, it is possible to build custom cloud storage management software stacks. With the application of RESTful APIs to a web service, desirable properties could be induced, such as performance, scalability, and modifiability, to enable services to work best on the Web.

For an example eCommerce web application (Amazon), a centralized GUI along with RESTful applications that are provided by the underlying storage infrastructure are used to provide monitoring and metering capabilities. The design of object storage also simplifies how users access data of eCommerce web application through the use of RESTful APIs, such as GET, PUT, and DELETE. Openstack Swift and the Amazon Simple Storage Service (S3) are specific examples of object storage. Also, a RESTful web service exposes a set of resources that identify the targets of the interaction with its clients. Resources are identified by URIs, which provide a global addressing space for resource and service discovery.

4. Scalability

Scalability is the ability to non-disruptively add capacity and remove it as needed in a global namespace is a key function for storage clouds. A global namespace aggregates disparate storage infrastructure, potentially across geographical boundaries, to provide a consolidated file or object view that simplifies administration

For an example eCommerce web application (Amazon), messaging systems can provide scalability. When using messaging tools, direct dependencies between services are removed, which improves reliability and scalability. The asynchronous relationship between saving and retrieving messages also provides scalability and reliability benefits. Amazon Simple Storage Service (Amazon S3) is an object storage service that offers industry-leading scalability.

Part2

A. Build a basic Django Web App for a very basic ecommerce use case, keep it simple, following the below references. [45 points]

1. home view: (example as below)

user: display username on the up right corner

products: show the product and description, add a button to add to cart (change the below projects data to products information)

2. shopping cart view:

Show a table of added products, user can do CRUD of the product.

Graphical user interface, application, website

Description automatically generated

B. Now inspect this fully functional eCommerce Web App using Django (from the reference below). How does this design compare with what you learned in Part I above? Can you use Django to extend your basic app from A to build this? (No need to implement or code. ONLY answer this part B using diagrams and your explanation.) [15 points]