CMR
INSTITUTE OF
TECHNOLOGY

USN | | | | | | | | | | |

Internal Assessment Test 3– April. 2025

Sub:	Cloud Computing							Sub Code:	22MCA332
Date:	8/4/2025	Duration:	90 min's	Max Marks:	50	Sem:	III	Branch:	MCA

Note : Answer FIVE FULL Questions, choosing ONE full question from each Module

		MARKS	OBE	
			CO	RBT
	PART I			
1	Explain with suitable diagram the cloud computing architecture.	[10]	CO4	L2
	OR			
2	Explain platform as a service reference model.	[10]	CO4	L2
	PART II			
3	Discuss Community cloud in detail	[10]	CO4	L2
	OR			
4	With a neat diagram, explain infrastructure as a service reference implementation	[10]	CO4	L2
	PART III			
5	Describe Google AppEngine Platform Architecture	[10]	CO4	L3
	OR			
6	Briefly explain EC2 instance of AWS	[10]	CO4	L3
	PART IV			
7	Discuss the storage services provided by windows Azure	[10]	CO4	L2
	OR			
8	Discuss the storage services offered by Amazon Web Services (AWS)	[10]	CO4	L2
	PARTV			
9	Briefly explain architecture and overview of the Jeeva Portal	[10]	CO4	L3
	OR			
10	Explain the salesforce and Force.com architecture .	[10]	CO4	L3

1. Explain with suitable diagram the cloud computing architecture

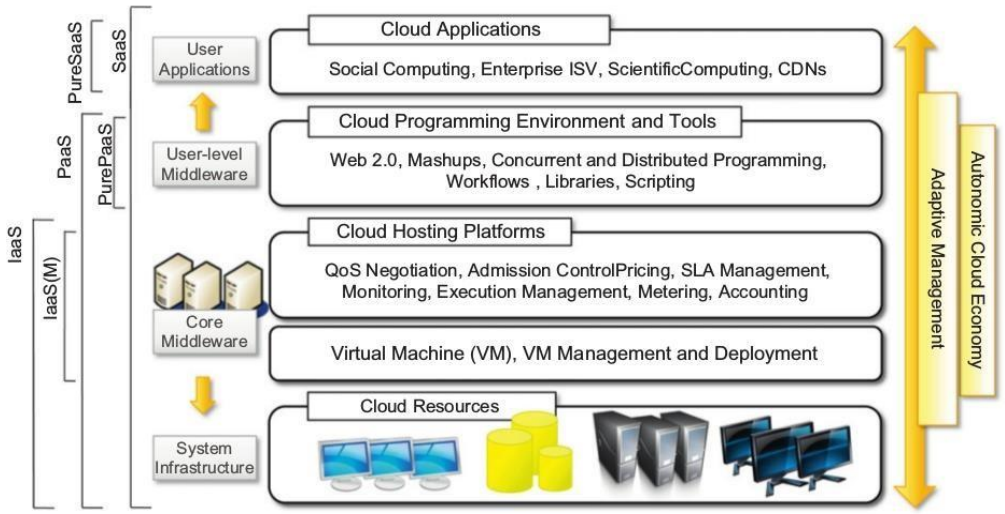

FIGURE 4.1
The cloud computing architecture.

It is possible to organize all the concrete realizations of cloud computing into a layered view covering the entire stack (see Figure 4.1), from hardware appliances to software systems. Cloud infrastructure can be heterogeneous in nature because a variety of resources, such as clusters and even networked PCs, can be used to build it.

- The **physical infrastructure** is managed by the **core middleware**, the objectives of which are to provide an appropriate **runtime environment** for applications and to best utilize resources.
- At the bottom of the stack, **virtualization technologies** are used to guarantee runtime environment customization, application isolation, sandboxing, and quality of service. **Hardware virtualization** is most used at this level. **Hypervisors** manage the pool of resources and expose the distributed infrastructure as a collection of virtual machines. By using virtual machine technology, it is possible tofinely partition the hardware resources such as CPU and memory and to virtualize specific devices,thus meeting the requirementsof users and applications. This solution is generally paired with storage and network virtualization strategies, which allow the infrastructure to be completely virtualized and controlled.
- **Infrastructure management** is the key function of core middleware, which supports capabilities such as negotiation of the quality of service, admission control, execution management and monitoring, accounting, and billing. The combination of **cloud hosting platforms and resources** is generally classified as an **Infrastructure-as-a-Service(IaaS)** solution.

- We can organize the different examples of IaaS into two categories: Some of them provide both the management layer and the physical infrastructure; others provide only the management layer (IaaS (M)).
- **User-level middleware:** The range of tools include Web-based interfaces, command-line tools, and frameworks for concurrent and distributed programming. In this scenario, users develop their applications specifically for the cloud by using the API exposed at the user-level middleware. For this reason, this approach is also known as **Platform-as-a-Service (PaaS)** because the service offered to the user is a **development platform** rather than an infrastructure.
- The top layer of the reference model is **User Application level:** These are referred as **Software-as-a-Service (SaaS).** In most cases these are Web-based applications that rely on the cloud to provide service to end users.
- The horsepower of the cloud provided by IaaS and PaaS solutions allows independent software vendors to deliver their application services over the Internet.

Table 4.1 Cloud Computing Services Classification

Category	Characteristics	Product Type	Vendors and Products
SaaS	Customers are provided with applications that are accessible anytime and from anywhere.	Web applications and services (Web 2.0)	SalesForce.com (CRM) Clarizen.com (project management) Google Apps
PaaS	Customers are provided with a platform for developing applications hosted in the cloud.	Programming APIs and frameworks Deployment systems	Google AppEngine Microsoft Azure Manjrasoft Aneka Data Synapse
IaaS/HaaS	Customers are provided with virtualized hardware and storage on top of which they can build their infrastructure.	Virtual machine management infrastructure Storage management Network management	Amazon EC2 and S3 GoGrid Nirvanix

2. Explain platform as a service reference model.

Platform-as-a-Service (PaaS) solutions provide a **development and deployment platform for running applications in the cloud**. They constitute the middleware on top of which applications are built. A general overview of the features characterizing the PaaS approach I sgiven in Figure 4.3.Application management is the core functionality of the middleware.

PaaS implementations provide **applications with a runtime environment** and do not expose any service for managing the underlying infrastructure. They **automate** the process of **deploying** applications to the infrastructure, **configuring** application components, **provisioning**, and configuring supporting technologies such as **load balancers** and **databases**, and **managing system** change based on **policies** set by the **user**.

The **core middleware** oversees managing the resources and scaling applications on demand or automatically, according to the commitments made with users.
From a **user point of view**, the **core middleware exposes interfaces that allow programming and deploying applications on the cloud**.These can be in the **form of aWeb-basedinterface** or in the form of **programming APIs and libraries**.

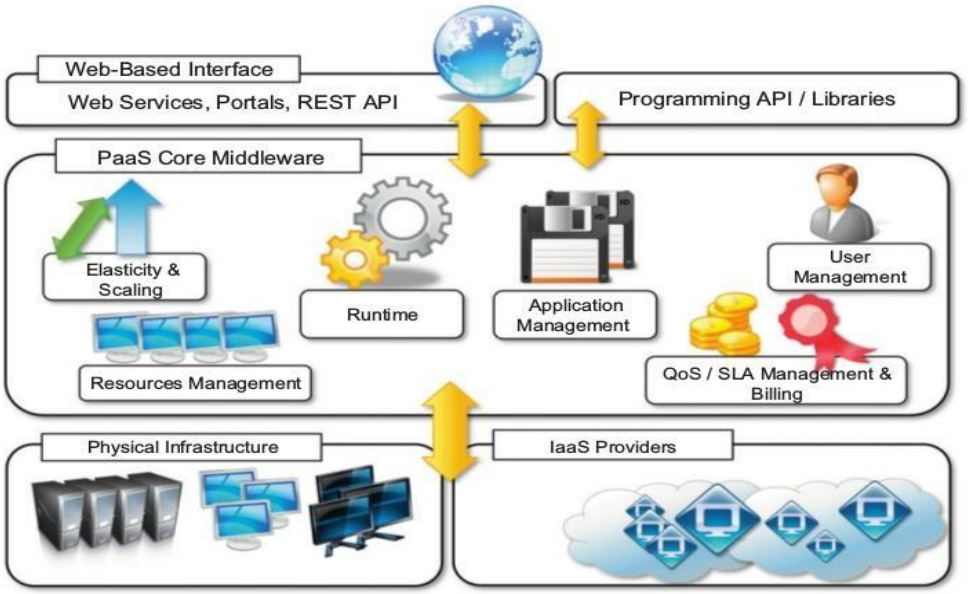

FIGURE 4.3
The Platform-as-a-Service reference model.

PaaS solutions can offer middleware for developing applications together with the infrastructure or simply provide users with the software that is installed on the user premises.
In the **first case**, the PaaS provider also owns large datacenters where applications are executed.
In the **secondcase**, referred as PurePaaS, the middle are constitutes the core value of the offering.

Table4.2 provides a classification of the most popular PaaS implementations. It is possible to organize the various solutions into three wide categories: PaaS-I, PaaS-II, and PaaS-III.

Table 4.2 Platform-as-a-Service Offering Classification

Category	Description	Product Type	Vendors and Products
PaaS-I	Runtime environment with Web-hosted application development platform. Rapid application prototyping.	Middleware + Infrastructure Middleware + Infrastructure	Force.com Longjump
PaaS-II	Runtime environment for scaling Web applications. The runtime could be enhanced by additional components that provide scaling capabilities.	Middleware + Infrastructure Middleware Middleware + Infrastructure Middleware + Infrastructure Middleware + Infrastructure Middleware	Google AppEngine AppScale Heroku Engine Yard Joyent Smart Platform GigaSpaces XAP
PaaS-III	Middleware and programming model for developing distributed applications in the cloud.	Middleware + Infrastructure Middleware Middleware Middleware Middleware Middleware	Microsoft Azure DataSynapse Cloud IQ Manjrasof Aneka Apprenda SaaSGrid GigaSpaces DataGrid

As noted by Sam Charrington, product manager at **Appistry.com**, there are some essential characteristics that identify a PaaS solution:

1. Runtime framework - This framework represents the "software stack" of the PaaS model. The runtime framework **executes end-user code** according to the policies set by the user and the provider.

2. Abstraction - PaaS, the focus is on the **applications the cloud must support**. PaaS solutions offer a way to **deploy and manage applications on the cloud** rather than a bunch of virtual machines on top of which the IT infrastructure is built and configured.

3. Automation-scaling them by provisioning **additional resources when needed**. This process is performed automatically and according to the SLA made between the customers and the provider.

4. Cloud services - PaaS offerings provide developers and architects with services and APIs, helping them to simplify the creation and delivery of elastic and highly available cloud applications.

3. Discuss Community cloud in detail

Community clouds are distributed systems created by integrating the services of different clouds to address the specific needs of an industry, a community, or a business sector.

The National Institute of Standards and Technologies (NIST) characterizes community clouds as follows:

"The infrastructure is shared by several organizations and supports a specific community that has shared concerns (e.g., mission, security requirements, policy, and compliance considerations). It may be managed by the organizations or a third party and may exist on premise or off premise."

Or

In the community cloud, the infrastructure is shared between organizations that have shared concerns or tasks. The cloud may be managed by an organization or a third party.

Figure4.6providesageneralviewoftheusagescenarioofcommunityclouds,togetherwith referencearchitecture.Theusersofaspecificcommunitycloudfallintoawell-identified

community, sharing the same concerns or needs; they can be government bodies, industries, oreven simple users, but all of them focus on the same issues for their interaction with the cloud.

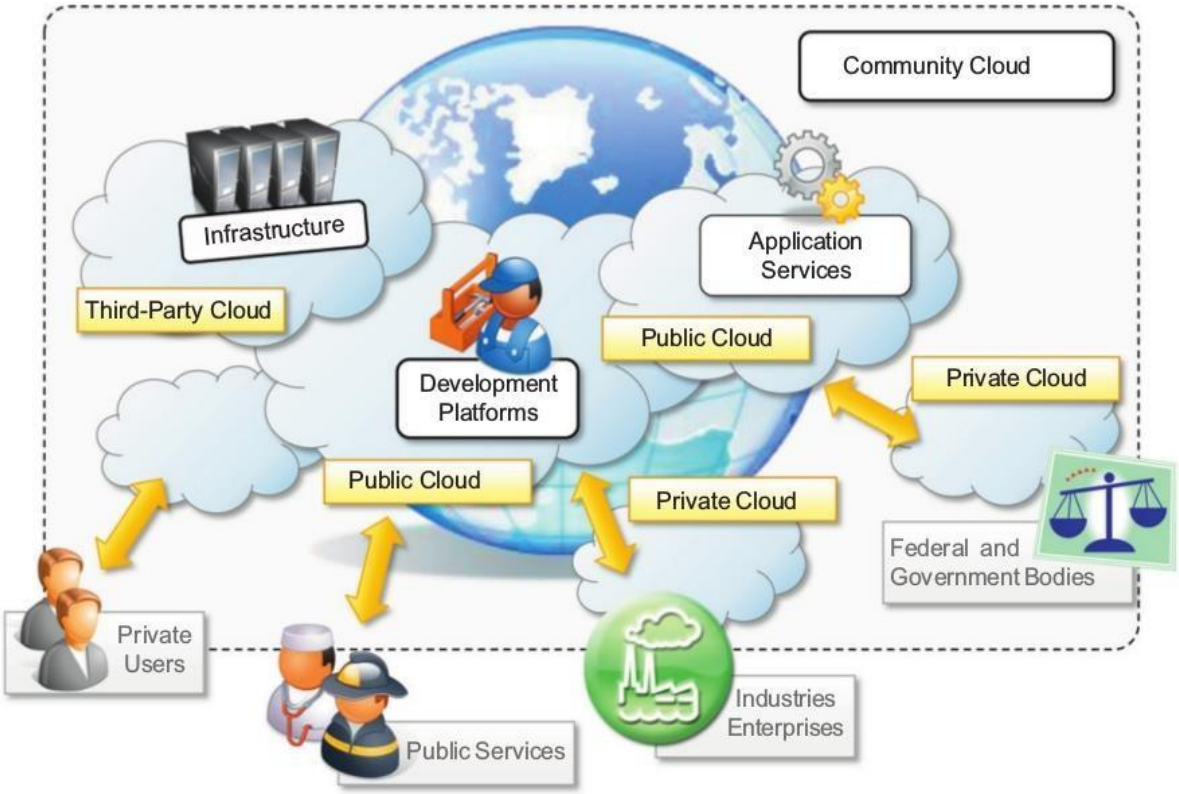

FIGURE 4.6

A community cloud.

Candidate sectors for community clouds areas follows:

1. **Media industry** - looking for low-cost, agile, and simple solutions to improve the efficiency of content production.
2. **Healthcare industry** - n the healthcare industry community clouds are used to share information and knowledge on the global level with sensitive data in the private infrastructure.
3. **Energy and other core industries -** In these sectors, community clouds can bundle the comprehensive set of solutions that together vertically address management, deployment, and orchestration of services and operations.
4. **Public sector-** Legal and political restrictions in the public sector can limit the adoption of public cloud offerings. Moreover, governmental processes involve several institutions and agencies and are aimed at providing strategic solutions at local, national, and international administrative levels.
5. **Scientific research -** In this organization with common interests in science share a large, distributed infrastructure for scientific computing.

The benefits of these community clouds are the following:

1. **Openness.** By removing the dependency on cloud vendors, community clouds are open systems in which fair competition between different solutions can happen.
2. **Community.** Being based on a collective that provides resources and services, the infrastructure

turns out to be more scalable because the system can grow simply by expanding its user base.

3. **Graceful failures.** Since there is no single provider or vendor in control of the infrastructure, there is no single point of failure.

4. **Convenience and control.** Within a community cloud there is no conflict between convenience and control because the cloud is shared and owned by the community, which makes all the decisions through a collective democratic process.

5. **Environmental sustainability.** The community cloud is supposed to have a smaller carbon footprint because it harnesses underutilized resources. Moreover, these clouds tend to be more organic by growing and shrinking in a symbiotic relationship to support the demand of the community, which in turn sustains it.

4. With a neat diagram, explain infrastructure as a service reference implementation

Infrastructure / Hardware-as-a-Service (IaaS / HaaS) solutions are the most popular and developed market segment of cloud computing. They **deliver customizable infrastructure on demand**.

The available options within the IaaS offering umbrella range from single servers to entire infrastructures, including network devices, load balancers, and database and Web servers.

The main technology used to deliver and implement these solutions is **hardware virtualization**: one or more virtual machines opportunely configured and interconnected define the distributed system on top of which applications are installed and deployed. Virtual machines also constitute the atomic components that are deployed and priced according to the specific features of the virtual hardware: memory, number of processors, and disk storage.

From the perspective of the customer, it reduces the administration and maintenance cost as well as the capital costs allocated to purchase hardware.

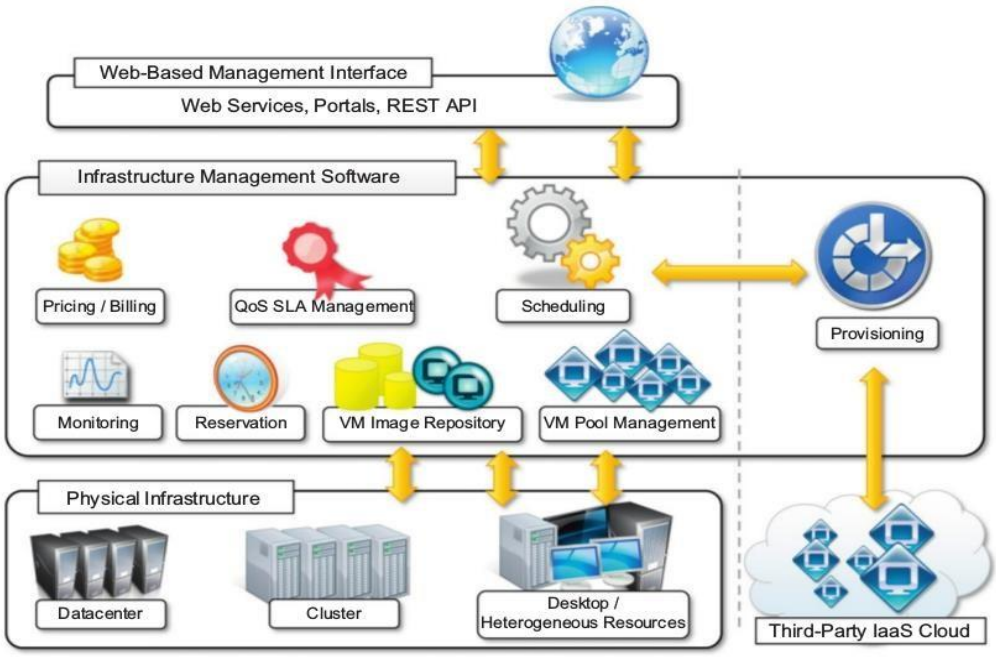

FIGURE 4.2
Infrastructure-as-a-Service reference implementation.

- At the top layer the **user interface (Web based Management Interface):** provides access to the services exposed by the software management infrastructure. Such an **interface is based** on**Web 2.0 technologies:Web services,RESTfulAPIs, and mashups**. These technologies allow either applications or final users to access the services exposed by the underlying infrastructure. Web 2.0 applications allow developing full- featured management consoles completely hosted in a browser or a Web page. Web services and RESTful APIs allow programs to interact with the service without human intervention, thus providing complete integration within a software system.

- The core features of an IaaS solution are implemented in the **infrastructure management software layer:** In particular, management of the virtual machines is the most important function performed by this layer. A central role is played by the scheduler, which oversees **allocating the execution of virtual machine instances**. The scheduler interacts with the other components that perform a variety of tasks:
  - The **pricing and billing** component takes care of the cost of executing each virtual machine instance and maintains data that will be used to charge the user.
  - The **monitoring component** tracks the execution of each virtual machine instance and maintains data required for reporting and analyzing the performance of the system.
  - The **reservation component** stores the information of all the virtual machine instances that have been executed or that will be executed in the future.
  - If support for **QoS-based** execution is provided, a QoS/SLA management component will maintain repository of all the SLAs made with the users; together with the monitoring component, this component is used to ensure that a given virtual machine instance is executed with the desired **quality of service**.
  - The **VM Image repository** component provides a catalog of virtual machine images that users can use to **create virtual instances**. Some implementations also allow users to upload their specific virtual machine images.
  - A **VM Pool Management** component is responsible for keeping track of all the live instances.
  - Finally, if the system supports the integration of additional resources belonging to a third-party IaaS provider, a **provisioning component** interacts with the scheduler to provide a **virtual machine instance** that is **external** to the local physical infrastructure directly managed by the pool.
- The bottom layer is composed of the **physical infrastructure**, on top of which the management layer operates. As previously discussed, the infrastructure can be of different types; the specific infrastructure used depends on the specific use of the cloud. A cloud infrastructure developed in house, in a small or medium-sized enterprise or within a university department, will most likely rely on a cluster. At the bottom of the scale, it is also possible to consider a heterogeneous environment where different types of resources—PCs, workstations, and clusters—can be aggregated.

5. Describe Google AppEngine Platform Architecture

Google AppEngine is a PaaS implementation that provides services for developing and hosting scalable Web applications. AppEngine is essentially a distributed and scalable runtime environment that leverages Google's distributed infrastructure to scale out applications facing a large number of requests by allocating more computing resources to them and balancing the load among them.

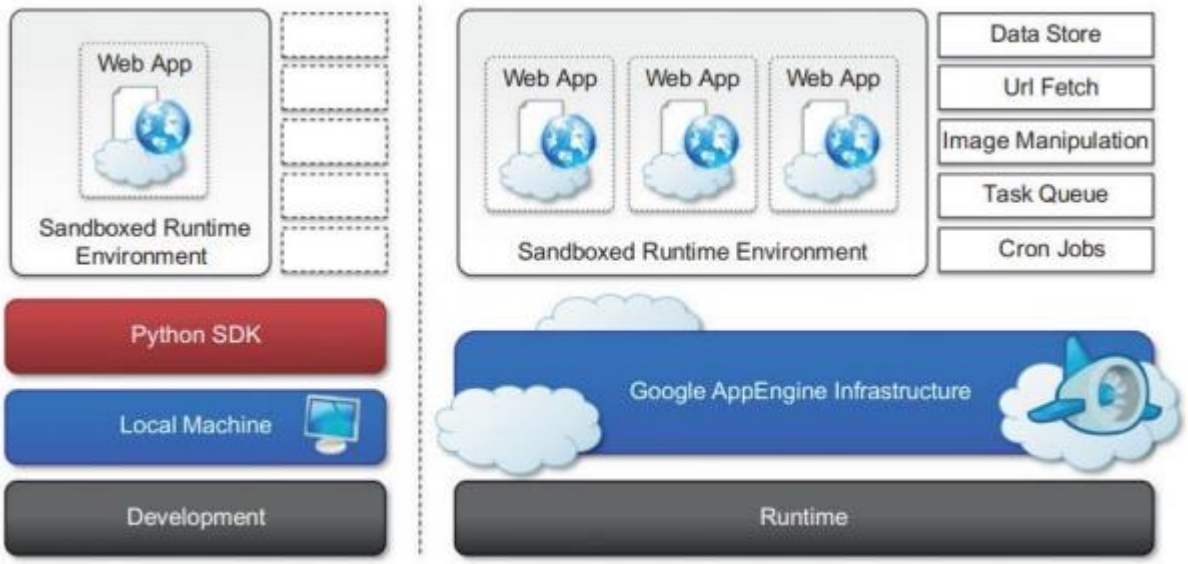

FIGURE 9.2

Google AppEngine platform architecture.

1) Infrastructure

AppEngine hosts Web applications, and its primary function is to serve users requests efficiently. To do so, AppEngine's infrastructure takes advantage of many servers available within Google datacenters.

2) Runtime environment

- The runtime environment represents the execution context of applications hosted on AppEngine. With reference to the AppEngine infrastructure code, which is always active and running, the runtime comes into existence when the request handler starts executing and terminates once the handler has completed.
- Sandboxing One of the major responsibilities of the runtime environment is to provide the application environment with an isolated and protected context in which it can execute without causing a threat to the server and without being influenced by other applications.
- Supported runtimes: Currently, it is possible to develop AppEngine applications using three different languages and related technologies: Java, Python, and Go. AppEngine currently supports Java 6, and developers can use the common tools for Web application development in Java, such as the Java Server Pages (JSP), and the applications interact with the environment by using the Java Servlet standard

3) Storage

AppEngine provides various types of storage, which operate differently depending on the volatility of the data. There are three different levels of storage: in memory-cache, storage for semistructured data, and long-term storage for static data.

- Static file servers : Web applications are composed of dynamic and static data. Dynamic data are a result of the logic of the application and the interaction with the user.
- DataStore is a service that allows developers to store semistructured data. The service is designed to scale and optimized to quickly access data. DataStore can be considered as a large object database in which to store objects that can be retrieved by a specified key. Both the type of the key and the structure of the object can vary.

4) Application services

Applications hosted on AppEngine take the most from the services made available through the runtime environment. These services simplify most of the common operations that are performed in Web applications: access to data, account management, integration of external resources, messaging and communication, image manipulation, and asynchronous computation.

- UrlFetch : The sandbox environment does not allow applications to open arbitrary connections through sockets, but it does provide developers with the capability of retrieving a remote resource through HTTP/HTTPS by means of the UrlFetch service.
- MemCache : AppEngine provides caching services by means of MemCache. This is a distributed in-memory cache that is optimized for fast access and provides developers with a volatile store for the objects that are frequently accessed.
- Mail and instant messaging: AppEngine provides developers with the ability to send and receive mails through Mail. The service allows sending email on behalf of the application to specific user accounts.
- Account management: AppEngine simplifies account management by allowing developers to leverage Google account management by means of Google Accounts.
- Image manipulation: AppEngine allows applications to perform image resizing, rotation, mirroring, and enhancement by means of Image Manipulation, a service that is also used in other Google products.

5) Compute services

AppEngine offers additional services such as Task Queues and Cron Jobs that simplify the execution of computations that are off-bandwidth or those that cannot be performed within the timeframe of the Web request.

- Task queues allow applications to submit a task for a later execution.
- Cron jobs : it is possible to schedule the required operation at the desired time by using the Cron Jobs service.

6. Briefly explain EC2 instance of AWS

   EC2 instances represent virtual machines. They are created using AMI as templates, which are specialized by selecting the number of cores, their computing power, and the installed memory

   - The processing power is expressed in terms of virtual cores and EC2 Compute Units (ECUs). The ECU is a measure of the computing power of a virtual core; it is used to express a predictable quantity of real CPU power that is allocated to an instance.
   - By using compute units instead of real frequency values, Amazon can change over time the mapping of such units to the underlying real amount of computing power allocated, thus keeping the performance of EC2 instances consistent with standards set by the times.
   - Over time, the hardware supporting the underlying infrastructure will be replaced by more powerful hardware, and the use of ECUs helps give users a consistent view of the performance offered by EC2 instances.
   - Since users rent computing capacity rather than buying hardware, this approach is reasonable
   - Six major currently available configurations categories for EC2 instances.:
     - Standard instances. This class offers a set of configurations that are suitable for most applications. EC2 provides three different categories of increasing computing power, storage, and memory.
     - Micro instances. This class is suitable for those applications that consume a limited amount of computing power and memory and occasionally need bursts in CPU cycles to process surges in the workload. Micro instances can be used for small Web applications with limited traffic.

- High-memory instances. This class targets applications that need to process huge workloads and require large amounts of memory.
- High-CPU instances. This class targets compute-intensive applications. Two configurations are available where computing power proportionally increases more than memory.
- Cluster Compute instances. This class is used to provide virtual cluster services. Instances in this category are characterized by high CPU compute power and large memory and an extremely high I/O and network performance, which makes it suitable for HPC applications.
- Cluster GPU instances. This class provides instances featuring graphic processing units (GPUs) and high compute power, large memory, and extremely high I/O and network performance..

- EC2 instances are priced hourly according to the category they belong to. At the beginning of every hour of usage, the user will be charged the cost of the entire hour. The hourly expense charged for one instance is constant.
- Another alternative is represented by spot instances. These instances are much more dynamic in terms of pricing and lifetime since they are made available to the user according to the load of EC2 and the availability of resources.
- EC2 instances can be run either by using the command-line tools provided by Amazon, which connects the Amazon Web Service that provides remote access to the EC2 infrastructure, or via the AWS console, which allows the management of other services, such as S3

7. Discuss the storage services provided by windows Azure

Windows Azure provides different types of storage solutions that complement compute services with a more durable and redundant option compared to local storage. Compared to local storage, these services can be accessed by multiple clients at the same time and from everywhere, thus becoming a general solution for storage.

1) Blobs

Azure allows storing large amount of data in the form of binary large objects (BLOBs) by means of the blobs service. This service is optimal to store large text or binary files. Two types of blobs are available:

• Block blobs. Block blobs are composed of blocks and are optimized for sequential access; therefore they are appropriate for media streaming. Currently, blocks are of 4 MB, and a single block blob can reach 200 GB in dimension.

• Page blobs. Page blobs are made of pages that are identified by an offset from the beginning of the blob. A page blob can be split into multiple pages or constituted of a single page. This type of blob is optimized for random access and can be used to host data different from streaming. Currently, the maximum dimension of a page blob can be 1 TB.

Blobs storage provides users with the ability to describe the data by adding metadata. It is also possible to take snapshots of a blob for backup purposes. Moreover, to optimize its distribution, blobs storage can leverage the Windows Azure CDN so that blobs are kept close to users requesting them and can be served efficiently.

2) Azure drive

Page blobs can be used to store an entire file system in the form of a single Virtual Hard Drive (VHD) file. This can then be mounted as a part of the NTFS file system by Azure compute resources, thus providing persistent and durable storage. A page blob mounted as part of an NTFS tree is called an Azure Drive.

3) Tables

Tables constitute a semi-structured storage solution, allowing users to store information in the form of entities with a collection of properties. Entities are stored as rows in the table and are identified by a key, which also constitutes the unique index built for the table. Users can insert, update, delete, and select a subset of the rows stored in the table. Unlike SQL tables, there are no schema enforcing constraints on the properties of entities and there is no facility for representing relationships among entities. For this reason, tables are more similar to spreadsheets rather than SQL tables. The service is designed to handle large amounts of data and queries returning huge result sets. This capability is supported by partial result sets and table partitions.

4) Queues

Queue storage allows applications to communicate by exchanging messages through durable queues, thus avoiding lost or unprocessed messages. Applications enter messages into a queue, and other applications can read them in a first-in, first-out (FIFO) style. To ensure that messages get processed, when an application reads a message it is marked as invisible; hence it will not be available to other clients. Once the application has completed processing the message, it needs to explicitly delete the message from the queue. This two-phase process ensures that messages get processed before they are removed from the queue, and the client failures do not prevent messages from being processed. At the same time, this is also a reason that the queue does not enforce a strict FIFO model: Messages that are read by applications that crash during processing are made available again after a timeout, during which other messages can be read by other clients. An alternative to reading a message is peeking, which allows retrieving the message but letting it stay visible in the queue. Messages that are peeked are not considered processed. All the services described are geo-replicated three times to ensure their availability in case of major disasters. Geo-replication involves the copying of data into a different datacenter that is hundreds or thousands of miles away from the original datacenter.

8. Discuss the storage services offered by Amazon Web Services (AWS)

1) Simple Storage Service (S3)
This is a distributed object store that allows users to store information in different formats.
The core components of S3 are two: buckets and objects. Buckets represent virtual containers in which to store objects; objects represent the content that is actually stored. Objects can also be enriched with metadata that can be used to tag the stored content with additional information. As the name suggests, S3 has been designed to provide a simple storage service that's accessible through a Representational State Transfer (REST) interface, which is quite similar to a distributed file system but which presents some important differences that allow the infrastructure to be highly efficient:

- The storage is organized in a two-level hierarchy.
- Stored objects cannot be manipulated like standard files
- Content is not immediately available to users
- Requests will occasionally fail. Due to the large distributed infrastructure being managed, requests for object may occasionally fail.

Buckets, objects, and attached metadata are made accessible through a REST interface. Therefore, they are represented by uniform resource identifiers (URIs) under the s3.amazonaws.com domain. All the operations are then performed by expressing the entity they are directed to in the form of a request for a URI. Amazon offers three different ways of addressing a bucket:

- Canonical form: http://s3.amazonaws.com/bukect_name/object_name
- Subdomain form: http://bucket-name/s3.amzonaws.com/object_name
- Virtual hosting form: http://bucket-name.com/object_name

Currently, five different permissions can be used:
• READ allows the grantee to retrieve an object and its metadata and to list the content of a bucket as well as getting its metadata.
• WRITE allows the grantee to add an object to a bucket as well as modify and remove it.
• READ_ACP allows the grantee to read the ACP of a resource.
• WRITE_ACP allows the grantee to modify the ACP of a resource.
• FULL_CONTROL grants all of the preceding permissions

2) Amazon elastic block store

- The Amazon Elastic Block Store (EBS) allows AWS users to provide EC2 instances with persistent storage in the form of volumes that can be mounted at instance startup.
- They accommodate up to 1 TB of space and are accessed through a block device interface, thus allowing users to format them according to the needs of the instance they are connected to (raw storage, file system, or other).
- The content of an EBS volume survives the instance life cycle and is persisted into S3. EBS volumes can be cloned, used as boot partitions, and constitute durable storage since they rely on S3 and it is possible to take incremental snapshots of their content..

3) Amazon ElastiCache

- ElastiCache is an implementation of an elastic in-memory cache based on a cluster of EC2 instances.
- It provides fast data access from other EC2 instances through a Memcached- compatible protocol so that existing applications based on such technology do not need to be modified and can transparently migrate to ElastiCache

4) Structured storage solutions

- Amazon provides applications with structured storage services in three different forms: preconfigured EC2 AMIs, Amazon Relational Data Storage (RDS), and Amazon SimpleDB.
- Preconfigured EC2 AMIs are predefined templates featuring an installation of a given database management system. EC2 instances created from these AMIs can be completed with an EBS volume for storage persistence. Available AMIs include installations of IBM DB2, Microsoft SQL Server, MySQL, Oracle, PostgreSQL, Sybase, and Vertica.
- Amazon RDS RDS is relational database service that relies on the EC2 infrastructure and is managed by Amazon.
- Amazon SimpleDB is a lightweight, highly scalable, and flexible data storage solution for applications that do not require a fully relational model for their data. SimpleDB provides support for semistructured data, the model for which is based on the concept of domains, items, and attributes.

5) Amazon CloudFront

- CloudFront is an implementation of a content delivery network on top of the Amazon distributed storage infrastructure. It leverages a collection of edge servers strategically located around the globe to better serve requests for static and streaming Web content so that the transfer time is reduced as much as possible.

Q9. Briefly explain architecture and overview of the Jeeva Portal

Applications in biology require high computing capabilities and operate on large data-sets that cause extensive I/O operations.

Therefore, biology applications have made use of supercomputing and cluster computing infrastructures.

Similar capabilities can be leveraged using cloud computing technologies in a more dynamic fashion, thus opening new opportunities for bioinformatics applications.

Protein structure prediction is a computationally intensive task that is fundamental to different types of research in the life sciences.

The geometric structure of a protein cannot be directly inferred from the sequence of genes that compose its structure, but it is the result of complex computations aimed at identifying the structure that minimizes the required energy.

This task requires the investigation of a space with a massive number of states, consequently creating many computations for each of these states.

One project that investigates the use of cloud technologies for protein structure prediction is Jeeva - an integrated Web portal that enables scientists to offload the prediction task to a computing cloud based on Aneka (Figure 10.2).

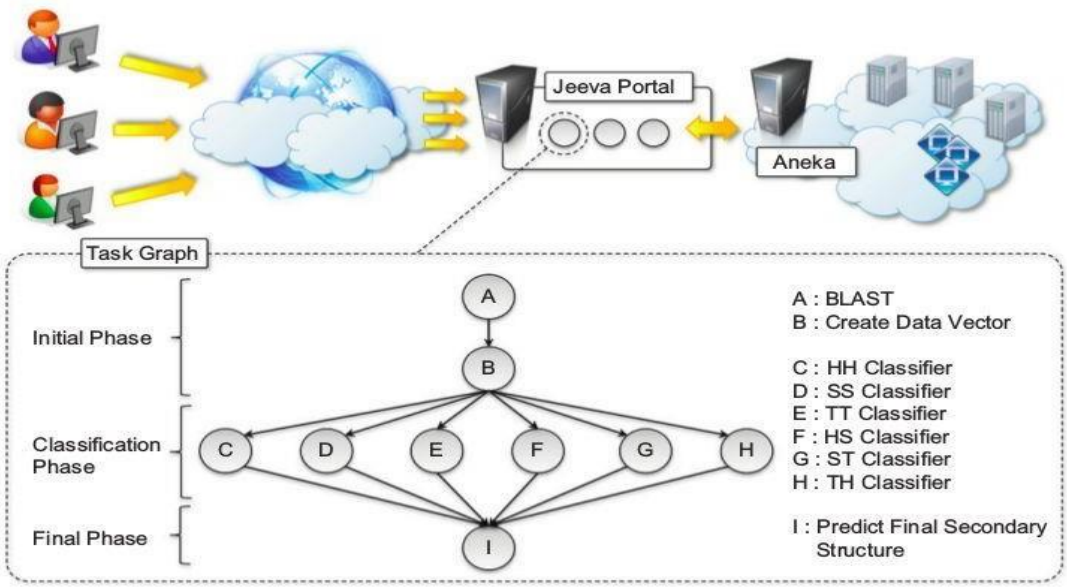

FIGURE 10.2

Architecture and overview of the Jeeva Portal.

The prediction task uses machine learning techniques (support vector machines) for determining the secondary structure of proteins. These techniques translate problem into one of pattern recognition,

where a sequence has to be classified into one of three possible classes (E, H, and C). A popular implementation based on support vector machines divides the pattern recognition problem into three phases: initialization, classification, and a final phase. These three phases have to be executed in sequence, we can perform parallel execution in the classification phase, where multiple classifiers are executed concurrently. This reduces computational time of the prediction. The prediction algorithm is then translated into a task graph that is submitted to Aneka. Once the task is completed, the middleware makes the results available for visualization through the portal.

The **advantage** of using cloud technologies is the capability to leverage a scalable computing infrastructure that can be grown and shrunk on demand.

9. Explain the salesforce and Force.com architecture

Salesforce.com is most popular and developed CRM solution available today. As of today more than 100,000 customers have chosen Safesforce.com to implement their CRM solutions. The application provides customizable CRM solutions that can be integrated with additional features developed by third parties.

Salesforce.com is based on the Force.com cloud development platform.

This represents scalable and high-performance middleware executing all operations of all Salesforce.com applications.

The architecture of the Force.com platform is shown in Figure 10.5. At the core of the platform resides its metadata architecture, which provides the system with flexibility and scalability.

Application core logic and business rules are saved as metadata into the Force.com store.

Both application structure and application data are stored in the store. A runtime engine executes application logic by retrieving its metadata and then performing the operations on the data.

A full-text search engine supports the runtime engine. This allows application users to have an effective user experience The search engine maintains its indexing data in a separate store.

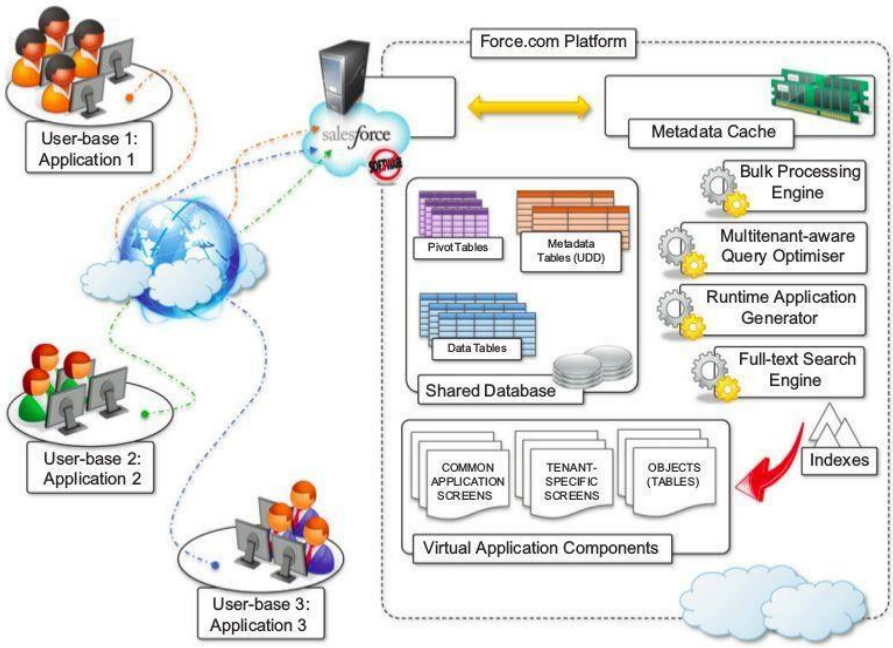

FIGURE 10.5

Salesforce.com and Force.com architecture.